Politecnico di Torino

Politecnico di Torino

Ingegneria Informatica

A.a. 2022/2023

Sessione di laurea Dicembre 2023

Canvas Coding: una piattaforma di gioco per studenti basata sulla programmazione creativa

Relatori:

Luigi De Russis

Juan Pablo Sáenz Moreno

Candidato:

Antonello Caputo

Sommario

Il Creative coding consiste nell'utilizzo della programmazione come mezzo di generazione di contenuti creativi. È un concetto in rapida e continua espansione che supera la distinzione tra arte e design, scienza ed ingegneria, comprendendo attività come l'arte generativa, editing audio e video, proiezioni, installazioni artistiche e tanto altro. La sua natura versatile, la possibilità di produrre espressioni artistiche ed il feedback visivo lo rendono un ottimo strumento per attrarre studenti e sviluppatori novizi al mondo della programmazione. Inoltre, la sua natura artistica e creativa lo rendono un ottimo strumento inclusivo che supera le barriere di genere, spesso imposte dagli approcci più tradizionali alla programmazione. Questo lavoro di tesi nasce con l'obiettivo di identificare le principali problematiche sperimentate dagli sviluppatori novizi durante i corsi introduttivi alla programmazione, al fine di progettare ed implementare uno strumento che prenda in prestito alcuni concetti del creative coding per appianare o risolvere del tutto le problematiche individuate. In virtù di tale scopo è stata progettata e sviluppata "Canvas Coding", una piattaforma web educativa che ha l'obiettivo di presentare i concetti cardine della programmazione imperativa e ad oggetti sotto forma di un gioco online, nel quale l'utente possa produrre degli elementi grafici 3D a partire dal codice da lui scritto e viceversa. Suddivisa in livelli, la piattaforma chiede all'utente di completare degli step utilizzando inizialmente solo la componente grafica 3D, aumentando man mano la complessità, sino a richiedere l'utilizzo del solo codice. In questo modo l'utente può prendere dimestichezza con la piattaforma e con i concetti presentati, dando allo stesso tempo sfogo alla propria creatività mediante la personalizzazione degli elementi disponibili. Per iniziare, è stato effettuato uno studio della bibliografia con lo scopo di individuare le principali problematiche riscontrate dagli studenti frequentanti corsi STEM di introduzione alla programmazione in classi K-12, CS0 e CS1. Inoltre, sono state individuate le soluzioni già esistenti a tal fine, basate sui concetti di Creative Coding, Creative Thinking e Creative Computation. Segue il ragionamento che sta dietro alla progettazione di Canvas Coding, la realizzazione dei prototipi e l'effettiva implementazione della piattaforma. È stato svolto, infine, un test di usabilità della piattaforma su un gruppo di studenti frequentanti il 4 e 5 anno in diversi Istituti Tecnici Tecnologici con indirizzo Informatica, residenti in diverse regioni Italiane. Il lavoro di tesi si conclude con la discussione dei risultati ottenuti durante questa fase di test.

Indice

Elenco delle figure

Capitolo 1

Introduzione

Durante la prima lezione di qualsiasi corso di programmazione, che sia questo tenuto in università o a scuola, il primo comando mostrato agli studenti è quasi sempre lo stesso: stampare sulla console la scritta *"Hello world"*. Questa affermazione rimane vera anche al variare degli anni, a seconda del linguaggio di programmazione più in voga: nei primi anni 90 era C, negli anni 2000 Java, adesso magari è Python, ma in ogni caso il risultato è sempre lo stesso, ovvero stampare la fatidica scritta. Per quale motivo però si utilizza ancora questo comando come primo esempio? Sicuramente perché risulta facilmente riproducibile dagli studenti, consente inoltre di mostrare la sintassi utilizzata dal linguaggio per richiamare una funzione, ma il vero motivo è perché consente di generare un output visivo all'esecuzione di un comando. In questo modo, lo studente può vedere come un'azione sconosciuta definita all'interno di un codice estraneo possa generare un testo formattato, a lui molto più familiare. Il comando agisce da tramite, mostrando subito allo studente un esempio concreto di utilizzo del codice, ovvero generare del testo all'interno di una finestra.

Se questo esempio aveva un impatto maggiore in passato sugli studenti, quando le interfacce grafiche erano molto semplici e composte quasi completamente da testo, adesso ha invece un impatto molto più limitato, in quanto questi sono abituati a ben altro. Videogiochi 3D complessi, applicazioni con effetti grafici, intelligenza artificiale, sono solo alcuni esempi delle realtà nelle quali sono immersi gli studenti ogni giorno, oramai difficilmente impressionabili da una semplice scritta generata all'interno di una finestra. Con qualche click o tap sullo schermo, grazie alle nuove tecnologie disponibili, gli studenti riescono a generare realtà complesse, a prescindere dalle competenze informatiche possedute.

Questa mancanza di innovazione, praticamente un ossimoro se affiancato al concetto di informatica, si ripresenta andando ad analizzare gli esempi utilizzati dai libri di testo o dai docenti durante i corsi di introduzione alla programmazione. Esempi sono gli algoritmi di ordinamento, la ricorsione, classi ed oggetti, le strutture

iterative, tutti presentati utilizzando ancora gli stessi esempi ed esercizi ormai da decenni[1].

Numerosi studi dimostrano che, nonostante le nuove generazioni di studenti siano completamente immerse nella tecnologia, queste percepiscono l'informatica come una tematica complessa, noiosa e poco creativa[2, 3]. In alcuni casi, questa percezione peggiora, spingendo studenti a definirla *tediosa, antisociale ed irrilevante.*[4] Studenti e studentesse che iniziano ad approcciarsi a questo settore, tendono a perdere molto velocemente interesse a causa di numerosi fattori tra cui proprio la mancanza di creatività, sino all'abbandono degli studi.[5] L'informatica inoltre è ancora percepita come un ambiente prettamente maschile, le ragazze percepiscono questa materia noiosa, esprimendo una forte avversione nei confronti del computer [1, 6].

Per combattere questo astio nei confronti della materia, è necessario definire dei metodi alternativi che catturino l'attenzione degli studenti durante i corsi di introduzione alla programmazione, degli strumenti che consentano allo stesso tempo di trasmettere conoscenza e di esprimere la creatività al fine di mantenere vivo il loro interesse: la programmazione creativa (o *Creative Coding*) risponde a queste esigenze.

Con *Creative Coding* si intende un approccio esplorativo e guidato dall'estetica, in cui gli studenti creano progetti visivi ed opere d'arte utilizzando il codice [7]: l'obiettivo della programmazione è focalizzato esclusivamente nella produzione di opere artistiche piuttosto che nella produzione di un risultato utile e funzionale. A partire da questo concetto è possibile individuare numerose sfaccettature e varianti, in particolare i concetti di *Computazione Creativa* [1] e *Creative Thinking* [2] definiscono degli obiettivi che risultano più vicini a questo lavoro di tesi.

Prendendo in prestito alcuni concetti dal creative coding e associandoli a degli obiettivi di apprendimento, è possibile definire delle attività di potenziamento della creatività che presentano un riscontro effettivo e tangibile nell'apprendimento degli studenti. Ad esempio, generare un feedback visivo sotto forma di componente grafico ottenuto come risultato della programmazione, rende l'ambiente di apprendimento più interessante e meno sterile ed aumenta la motivazione degli studenti [2].

1.1 Obiettivo

L'obiettivo della tesi è progettare ed implementare una piattaforma di gioco online che utilizza la programmazione creativa al fine di risolvere le principali problematiche e difficoltà sperimentate dagli studenti durante i corsi di introduzione alla programmazione nelle scuole superiori e nei primi anni di università.

La prima fase di studio è stata svolta interamente mediante l'analisi degli articoli e delle pubblicazioni disponibili inerenti alle difficoltà e limitazioni sperimentate dagli

studenti durante i corsi di informatica in classi K12, CS0/1/2. Successivamente, è stato condotto uno studio dei fattori che influenzano l'apprendimento degli studenti, soffermandosi principalmente sulla *motivazione*: a partire da questo, un'analisi approfondita degli strumenti e delle tecniche che la rafforzano è stata condotta, in cui è emerso che l'utilizzo delle arti visive e della creatività sono degli ottimi strumenti per motivare e attirare l'interesse degli studenti.

A seguire, sono stati analizzati i concetti di creatività nel contesto educativo [8], in particolar modo nell'insegnamento dell'informatica: in questa fase sono emersi i primi riferimenti al Creative Coding, Creative Thinking e computazione creativa, che sono stati successivamente analizzati in dettaglio singolarmente. In questa fase di studio conclusiva sono quindi riassunti tutti gli strumenti e le tecniche già esistenti individuate: esempi sono Gamification e Robots [4, 9], Linguaggi di programmazione visivi [2, 10], Computational Creativity Exercises [11] e Ambienti di programmazione live [12].

Infine, sono stati analizzati i risultati dell'utilizzo degli strumenti e tecniche individuati nella fase di analisi; in particolare, è emerso:

- miglioramento dell'apprendimento dei concetti trattati durante i corsi di informatica [3];
- aumento del tempo di studio aggiuntivo, passato per puro divertimento e senza alcun obbligo da parte dei docenti [13, 4, 14];
- diminuzione dei tassi di abbandono scolastico e di fallimento della carriera universitaria [2].

Sulla base dei dati raccolti durante la fase di ricerca, si è ritenuto opportuno proseguire con la realizzazione del progetto di tesi: in particolare, è stato realizzato un gioco online chiamato **Canvas Coding** che consente agli utenti di interagire con componenti grafici 3D e con il codice, simultaneamente. Suddiviso in livelli, il gioco indica all'utente le operazioni da svolgere utilizzando i componenti grafici, gli strumenti disponibili, il codice oppure un mix di questi. L'utente può quindi:

- scrivere e compilare il codice Java visualizzando a schermo il risultato delle azioni svolte mediante il codice sotto forma di modifiche ai componenti grafici coinvolti
- utilizzare gli strumenti disponibili per generare e modificare i componenti grafici, visualizzando contemporaneamente l'evolversi del codice ad ogni azione svolta.

Suddiviso in step, ogni livello è focalizzato sull'apprendimento di una serie di nozioni teoriche e pratiche di programmazione: a partire dal programma scolastico di informatica insegnato in un istituto tecnico superiore, sono state estrapolate le

principali nozioni teoriche incentrate sulla programmazione imperativa e ad oggetti. Queste sono state associate agli step di ogni livello, disposti progressivamente in ordine crescente di difficoltà. In questo modo, gli studenti possono apprendere mentre giocano, passando in rassegna la maggior parte dei concetti insegnati in classe, in modo creativo ed innovativo.

Una volta completato lo sviluppo del gioco, è stato svolto uno studio di usabilità grazie alla partecipazione di alcuni studenti frequentanti il quarto e quinto anno in istituti tecnici e scientifici ad indirizzo informatico: i risultati ottenuti hanno confermato il trend individuato nella fase di analisi, ovvero che l'utilizzo della programmazione creativa durante i corsi di introduzione alla programmazione consente di migliorare l'apprendimento dei concetti trattati ed aumentare l'interesse degli studenti nei confronti dell'informatica.

1.2 Struttura della tesi

Nel **Capitolo 2** viene presentata la definizione allo stato dell'arte del concetto di Creative Coding e di creatività nel contesto informatico, a partire dai primi tentativi di utilizzo nel mondo della programmazione. Successivamente sono presentati i concetti derivati dal precedente, come il *Creative Computation*, *Computational Thinking* e *Creative Thinking*: per ognuno di questi viene fornita una definizione, degli esempi ed infine viene presentato un confronto. A seguire, è presentato il concetto di creatività applicata al contesto educativo: come incoraggiarne l'utilizzo all'interno di esercizi, quali strumenti attualmente utilizzati funzionano e come insegnare agli studenti mediante l'utilizzo della creatività. Per concludere, sono presentati e descritti una serie di strumenti e tecniche che utilizzano il Creative Coding, suddivisi in base ai campi di utilizzo: *Computational Music*, *Visual Arts*, *Coding and Media*.

Nel **Capitolo 3** sono delineate le scelte di progettazione che hanno influenzato lo sviluppo di Canvas Coding, partendo dall'identificazione dei requisiti e delle limitazioni degli strumenti didattici basati sul Creative Coding, individuati nel capitolo precedente. Tre proposte sono emerse da questo processo: un'estensione per un editor di codice, un'applicazione e un gioco. Per ciascuna di esse, sono esposti i requisiti soddisfatti, le limitazioni superate e gli aspetti positivi e negativi legati all'implementazione proposta. La scelta finale è ricaduta sullo sviluppo del gioco, motivo che verrà spiegato in dettaglio. Successivamente, vengono definite le caratteristiche principali del gioco, come la suddivisione in livelli e step, e come questi sono integrati con i concetti insegnati nel corso di informatica presso un istituto tecnico superiore. Il capitolo si conclude con la presentazione dei prototipi e una panoramica dettagliata delle principali funzionalità del gioco.

Nel **Capitolo 4** viene esaminata la scelta delle tecnologie adottate per la realizzazione del gioco Canvas Coding. Per ciascuna tecnologia, è presentata una panoramica dello stato attuale, sono esposti i motivi che ne hanno guidato la scelta e sono forniti degli esempi tratti direttamente dal codice di Canvas Coding. Successivamente, l'analisi si concentra sull'implementazione del gioco, distinguendo tra front-end e back-end: per entrambe le categorie, sono presentati dei frammenti di codice relativi all'implementazione di specifici componenti e funzionalità, seguiti da spiegazioni dettagliate degli aspetti tecnici e, ove possibile, arricchiti da immagini esplicative. Questo approfondimento mira a fornire una visione completa e dettagliata delle scelte tecnologiche e della fase di sviluppo del gioco.

Nel **Capitolo 5** sono presentati i risultati dei test di usabilità condotti su un gruppo di 5 studenti frequentati diverse scuole superiori ad indirizzo informatico, di diversa età e in diverse regioni d'Italia. Il capitolo espone nel dettaglio tutte le fasi precedenti all'esecuzione dei testi effettivi: a partire dalla fase di pianificazione, sono indicate tutte le scelte effettuate riguardanti i partecipanti, ruoli, strumenti, definizione dei task, metodologie da adoperare e metriche da utilizzare al fine di valutarne i risultati. A seguire, la fase di esecuzione dei test vera e propria consente di produrre una serie di dati che sono presentati nel dettaglio, per ogni partecipante del test. Infine, sono presentati i risultati ottenuti dalla fase di analisi, ricavati dall'utilizzo delle metriche prefissate.

Infine, nel **Capitolo 6** sono discussi i risultati ottenuti e sono presentati degli spunti per eventuali sviluppi futuri.

Capitolo 2

Background e Stato dell'Arte

Nel prima sezione è presentato il concetto di Creative Coding, a partire dalla fase embrionale sino allo stato dell'arte, mettendo in luce le sue caratteristiche principali e offrendo una serie di esempi e applicazioni pratiche nella vita di tutti i giorni. A seguire, sono dettagliati i concetti di creatività, Creative Thinking e Computational Thinking, ed un confronto tra questi è presentato al fine di chiarirne le differenze. Nel paragrafo successivo è delineato il ruolo della creatività nel settore educativo, presentando il trend attuale basato sui risultati ottenuti dagli articoli analizzati; inoltre sono esposte teorie ed approcci utilizzati nelle scuole ed università. Successivamente, vengono esaminati i principali framework e strumenti utilizzati per implementare il Creative Coding: Processing, P5, openFrameworks, seguito da un breve approfondimento sul Computational Music e sugli strumenti come Max e VVVV. Infine, sono presentati i risultati di apprendimento e le limitazioni individuate, derivati dall'analisi degli articoli presi in esame.

2.1 Creative Coding

Il **Creative Coding** (o Computazione Creativa) è una disciplina emergente che unisce teorie e metodologie dell'informatica e dell'ingegneria ai principi estetici, pratici e pedagogici delle arti grafiche. Utilizzando un approccio esplorativo e guidato dall'estetica, gli studenti costruiscono iterativamente progetti visivi ed opere d'arte, espandendo i loro progetti[7]. In poche parole, è la disciplina delle arti visive realizzate attraverso il codice. La Computazione Creativa, d'ora in avanti abbreviato con CC, richiede agli studenti di impegnarsi in atti creativi ed artistici, al fine di poter padroneggiare concetti essenziali nel contesto delle arti, ma anche nella definizione di algoritmi e codice[4]. Oltre ai principi di programmazione, gli

studenti apprendono operazioni grafiche essenziali come trasformazioni, iterazione e randomizzazione, disegno algoritmico, animazioni, fisica di base e interattività[1].

Il pioniere di questo approccio alla programmazione creativa è Jhon Maeda, informatico ed artista, che a partire dal 1999 ha radicalmente ri-contestualizzato il concetto di codice[4]. Lui ed il suo gruppo hanno esplorato la programmazione da un punto di vista pedagogico innovativo, applicandola per la prima volta al contesto delle arti insegnate nelle aule scolastiche. I risultati dei suoi primi studi condotti nei laboratori multimediali del MIT[1] sono raccolti all'interno del suo libro *Design By Numbers* [15], in particolare nella sezione *Aesthetics + Computation Group*. Il libro si basa sull'ipotesi che i progettisti possono utilizzare il codice come mezzo di espressione efficace di idee e concetti di design.

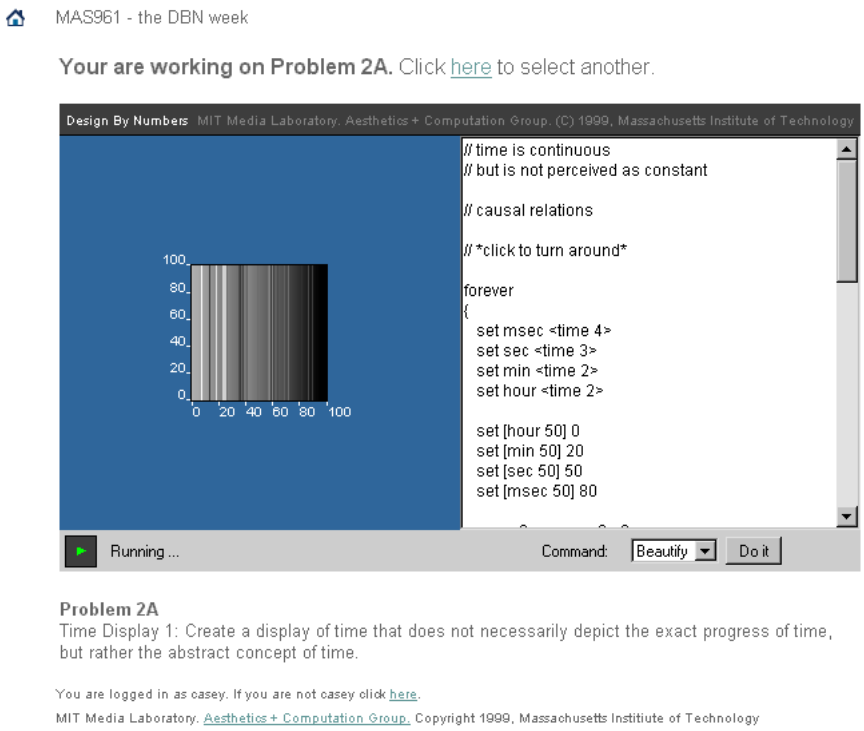

Figura 2.1: IDE Design By Numbers 1999

Secondo Maeda [4], conoscere alcuni concetti di programmazione e l'utilizzo di un linguaggio visivo, possono aiutare i designer ad esprimere meglio le loro idee, oltre che a fornire loro strumenti per realizzare progetti in modo innovativo

[1]Massachusetts Institute of Technology

e dinamico. Attraverso esempi pratici e concetti di programmazione, *Design by Numbers* mira a fornire una prospettiva nuova e creativa su come il design e la tecnologia possono interagire e influenzarsi reciprocamente, offrendo un approccio pratico per i designer per comunicare attraverso il linguaggio visivo. Nell'immagine 2.1 è presentata l'implementazione dell'ambiente di programmazione omonimo, precursore del moderno Processing[2], sviluppato dal team di Maeda.

All'interno del successivo libro *Creative Code*[16], l'autore si focalizza sull'utilizzo del codice come strumento creativo e sul modo in cui la programmazione può essere usata per generare opere d'arte e design innovativi, incentrandosi maggiormente sulla creatività nell'utilizzo del codice per produrre opere artistiche e di design.

Grazie alla sua natura versatile e dinamica, gli ambiti di applicazione del CC sono molteplici e ricoprono diversi settori, come:

1. **Arte interattiva**: creazione di installazioni artistiche audiovisive interattive che coinvolgono gli spettatori. Esempi sono progetti di arte digitale, opere d'arte generative che cambiano nel tempo o in base alle interazioni con il pubblico;

2. **Giochi**: il CC viene utilizzato per sviluppare grafica e nuove modalità di interazione e coinvolgimento degli utenti, ad esempio l'utilizzo di Robot [4];

3. **Design**: creazione di modelli, forme e strutture uniche che si adattano attraverso l'utilizzo di algoritmi e processi computazionali. Un esempio sono gli *E-textiles*, ovvero dei tessuti elettronici sintetici che cambiano colore, lunghezza o proprietà[17];

4. **Educazione**: numerosi sono gli esempi di applicazione nel contesto educativo e gli strumenti didattici sviluppati a supporto; linguaggi di programmazione visivi come Scratch [3] [18] o Projection Boxes [19], utilizzo del computer-vision [20] a fini educativi come ProtoObject [21] e tanti altri che verranno analizzati più nel dettaglio in seguito in quanto inerenti al tema trattato nella tesi;

5. **Media**: consente agli artisti di integrare elementi digitali e interattivi nella realizzazione delle proprie opere. Esempi noti di piattaforme a supporto sono P5.js [4], Media computation, Processing [5] e Gimp [6] [22].

[2] `www.processing.org`

[3] `www.scratch.mit.edu`

[4] `www.p5js.org`

[5] `www.processing.org`

[6] `www.gimp.org`

Per riassumere, le principali caratteristiche che contraddistinguono il Creative Coding sono le seguenti:

- **Creatività**: consente ad artisti, designer e creatori digitali di utilizzare il codice come mezzo di espressione della propria creatività al fine di generare contenuti unici e dal design innovativo[3];
- **Interattività**: grazie all'utilizzo della programmazione, consente di trasformare opere d'arte statiche in opere dinamiche, con le quali l'utente può interagire;
- **Accessibilità**: rende l'arte e la programmazione accessibile a tutti coloro che non hanno le competenze necessarie ma sono interessati ad approfondire l'argomento;
- **Condivisione**: una grande community di utenti è nata nel tempo, favorendo la condivisione di codice, risorse, progetti e conoscenza.

2.2 Creative thinking e Computational thinking

A partire dal concetto di Creative Coding o Creative Computation, numerosi altri concetti si sono diramati e specializzati in settori differenti. Questi, seppur sovrapponibili, presentano delle differenze distintive che sono analizzate nel dettaglio in seguito.

2.2.1 Creatività

Prima di dettagliare il Creative Thinking e Computational Thinking, è necessario partire dalla definizione del concetto di partenza: la creatività.

Per creatività, si intende la capacità di creare qualcosa di diverso, nuovo e innovativo utilizzando l'intelletto e la fantasia, immaginare nuove idee originali che coinvolgano innovazioni radicali.[23] Altre correnti di pensiero definiscono la creatività come l'applicazione di concetti già esistenti, ma pur sempre in modo innovativo.

La definizione di creatività che si avvicina di più al modo in cui questa è percepita nel mondo dell'informatica è una fusione di queste prospettive. Come citato da Salgian [24], la creatività nel mondo informatico è *"l'interazione tra attitudine, processo e ambiente attraverso il quale un individuo o un gruppo produce un prodotto tangibile, allo stesso tempo nuovo ed utile all'interno di un contesto sociale"*. Questa definizione si sposa bene con la percezione di creatività nel mondo informatico, dove essere creativi coinvolge la generazione di soluzioni utili e adattive a nuovi problemi da risolvere[2]. Romeike[18] definisce la creatività come *"qualcosa che*

porta a idee, soluzioni o intuizioni personali, nuove, uniche ed utili", sostenendo che informatica e creatività sono profondamente connessi.

E' necessario riconoscere la creatività, non come una forma di intelligenza innata, ma piuttosto come una *life skill* [7] essenziale che può essere insegnata e coltivata in tutti gli studenti.[25] La creatività nel campo dell'informatica è diversa da quella nelle arti, in quanto più facile da misurare, valutare e quindi da insegnare.

2.2.2 Definizioni

Il **Computational Thinking**, o Pensiero Computazionale, si riferisce ad un approccio analitico nella risoluzione dei problemi e nello sviluppo di soluzioni che utilizzano i principi di pensiero tipici dell'informatica[23]. Include abilità come: astrazione e generalizzazione di concetti secondo molteplici livelli, pattern recognition [8], design ed analisi di algoritmi, decomposizione di problemi in sotto componenti relazionali per migliorarne la definizione e la comprensione, valutazione delle soluzioni dei problemi al fine di individuare le più corrette, efficaci e di qualità[11]. L'obiettivo principale è utilizzare metodi computazionali per risolvere problemi, indifferentemente se questi siano legati all'informatica o ad altri campi.

La teoria della Generatività di Epstein[26] afferma che il pensiero creativo si basa su 4 competenze fondamentali, possedute da tutti e migliorabili attraverso la pratica:

1. catturare la novità
2. sfidare i modelli di pensiero e di comportamento consolidati
3. ampliare le proprie conoscenze
4. circondarsi di nuovi stimoli.

Il Computational Thinking può essere allenato attraverso un'istruzione "potenziata" dalla tecnologia in diverse aree, come programmazione o robotica. Nonostante sia strettamente collegata alle discipline scientifiche STEM [9], questa rappresenta un'abilità fondamentale che tutti devono possedere, non solo informatici o scienziati [27].

Nonostante possa sembrare un concetto incentrato solo sul singolo individuo, Sawyer [28] sostiene che la creatività sia invece un processo collaborativo, che coinvolge il pensiero collettivo. Questa visione afferma che il Creative Thinking

[7] Abilità utile nella vita quotidiana

[8] Riconoscimento di forme o configurazioni complesse

[9] Science, Technology, Engineering, Mathematics.

si verifica quando le persone lavorano collettivamente al fine di raggiungere dei progressi disciplinati su un problema, e le idee innovative emergono solo dopo la discussione e il dibattito tra diversi pensatori. Per questo motivo, la creatività nelle scuole dovrebbe essere insegnata attraverso opportunità di collaborazione ed "improvvisazioni disciplinate"[2].

Il **Creative Thinking**, o Pensiero Creativo, è un concetto più ampio che non è strettamente legato alla scienza: consiste nello sfruttare la novità e l'ingegno nella risoluzione di problemi computazionali, migliorandone l'efficacia e la qualità. È un processo mentale che porta a idee originali, soluzioni innovative e approcci non convenzionali alla risoluzione dei problemi. Ad esempio, una sfida può portare all'utilizzo di nuovi approcci nel modo in cui vengono adoperati gli strumenti computazionali, dando luogo a nuove prospettive su problemi esistenti; ampliare la propria conoscenza di base può introdurre concetti e procedure da altri domini per generare soluzioni computazionali[11].

La capacità di pensare in modo creativo è considerata una *soft skill* [10] significativa per lavorare nel settore dell'informatica, in quanto incoraggia la flessibilità, la fantasia, l'apertura mentale e la capacità di trovare connessioni inaspettate tra concetti o idee diverse.

2.2.3 Differenze

Nonostante questi concetti siano spesso sovrapponibili, è possibile individuare delle differenze sostanziali che li contraddistinguono. In particolare:

- **Computational Thinking** si focalizza maggiormente sull'approccio risolutivo tipico delle materie STEM;
- **Creative Coding** incorpora gli elementi creativi nella programmazione ai fini dell'esplorazione di nuove forme di espressione;
- **Creative Thinking** è un concetto più generico che si applica a diverse discipline, incoraggia l'innovazione e l'originalità nella risoluzione dei problemi.

2.3 Creatività nel contesto educativo

La creatività è la chiave per la produttività, l'adattabilità e l'efficienza. Partendo da questo punto di vista, Spendlove[25] afferma che *"la creatività non dovrebbe essere considerata opzionale o a scapito di altri obiettivi educativi, ma dovrebbe essere intrecciata ed incorporata in tutte le attività"*.

[10] Competenza trasversale.

Fino a qualche anno fa, però, la corrente di pensiero era ben diversa: l'arte e la creatività erano tenute a debita distanza dalle materie scientifiche. Questo approccio trova delle testimonianze già a partire dal 1959, quando C.P. Snow ha definito il concetto delle "Due Culture", riferendosi alle arti/scienze umanistiche "tradizionali" e alle materie scientifiche. Snow descrive quindi la netta separazione, le incomprensioni, le "possibilità creative" perse, la rigidità sociale delineate tra queste discipline sempre più separate e distinte, denunciando gli abissi incolmabili tra le scienze e le non-scienze. Sebbene il concetto delle Due Culture abbia avuto numerosi sostenitori, nell'ultimo mezzo secolo è stato spesso denigrato, infatti la "battaglia" tra discipline artistiche e scientifiche è ancora accesa[8].

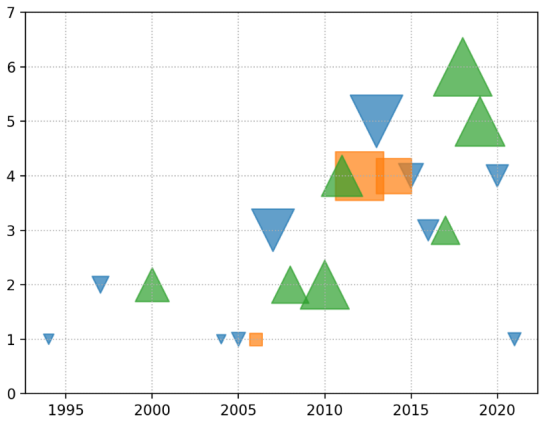

Figura 2.2: Scatter plot dei paper analizzati per anno, la dimensione dei triangoli è proporzionale al numero di istanze 'creativ*' trovate. Fonte immagine [3]

Intraprendere percorsi formativi e professionali inerenti alle discipline STEM è percepito dalla società come un approccio solido e pratico per immettersi successivamente nel mondo del lavoro. Negli ultimi anni si è insistito per includere le discipline legate all'arte, rappresentate dalla lettera "A", in questo modello educativo: alcuni insistono sul fatto che queste siano già incluse nelle materie STEM, altri invece credono che questa aggiunta possa causare distrazione all'interno del complesso sistema di insegnamenti scientifici, ormai già consolidato. Tuttavia, i sostenitori di una forma di apprendimento più inclusiva, indicano come l'aggiunta della "A" alle materie STEM, definendo le cosi dette STEAM[11], possa indirizzare gli studenti alle materie scientifiche ed artistiche, anche attraverso progetti pratici intrisi di opportunità creative. Per questo motivo, siccome la creatività può essere insegnata ed appresa, i curriculum scolastici devono essere ridefiniti al fine di

[11]Science, Technology, Engineering, Arts, Mathematics

integrare strumenti di insegnamento ed apprendimento volti a sviluppare capacità creative[2].

Negli ultimi decenni, i curriculum scolastici hanno iniziato a considerare le competenze non tecniche a capo di importanti risultati di apprendimento per i programmi di informatica ed ingegneria. Una recente analisi della letteratura sui corsi di introduzione alla programmazione ha rivelato che la creatività viene affrontata in questo contesto [13], anche se questo trend non è molto diffuso, nonostante lo sviluppo software sia comunque inteso come un processo creativo e collaborativo[3]. Come mostrato in figura 2.2, la ricerca della creatività nell'educazione informatica è aumentata notevolmente negli ultimi dieci anni, tuttavia, il volume di articoli incontrati risulta sorprendentemente basso se confrontato ad altri studi.

2.3.1 Teorie ed approcci

Dall'analisi degli articoli condotta, si evince che le principali tematiche che affrontano il concetto di creatività sono le seguenti:

- **Insegnare in K-12, CS-0/1/2**: l'attenzione principale è rivolta ai corsi di introduzione alla programmazione, in particolare l'utilizzo della creatività per migliorare l'insegnamento e l'apprendimento degli studenti [20, 4, 21, 3, 18, 1];
- **Computational Creativity**: concetto che unisce il "Computational Thinking" ed il "Creative Thinking" visti nella sezione precedente, per realizzare degli esercizi che migliorino le capacità di pensiero computazionale, le competenze creative e l'efficacia degli studenti [11];
- **Ambiente creativo**: l'obiettivo è la creazione di un "ambiente creativo" all'interno e all'esterno della classe. Ad esempio, offrire delle sale attrezzate per svolgere attività legate al Game Design e fornire degli strumenti che consentano di allenare il lato creativo anche all'esterno delle mura scolastiche[4, 3, 2];
- **Insegnare esplicitamente la creatività**: focalizzato principalmente sulla creatività, piuttosto che sul suo utilizzo come strumento per raggiungere obiettivi formativi. Per questo motivo, i computer sono intesi semplicemente come un veicolo che consente di esplorare ed esprimere la creatività [8, 22];
- **Motivazione**: Le opportunità di pensiero creativo nell'informatica sono fortemente influenzate dalla motivazione [18]. Uno studio condotto su programmatori che lavorano a dei progetti Open Source [12] ha dimostrato che la

[12]Il cui codice sorgente è pubblicamente accessibile.

motivazione per lavorare ad un progetto è influenzata dal livello di creatività concesso al programmatore rispetto al compito da eseguire. Allo stesso modo, gli studenti di informatica sperimentano una maggiore motivazione e divertimento quando viene permesso loro di essere creativi durante questi corsi[2, 21, 13].

A partire da questi temi, numerosi strumenti e tecniche sono stati sviluppati.

Uno dei più utilizzati è **l'Experimental Learning** [3], ovvero apprendimento attraverso esperimenti: mediante l'utilizzo di progetti ed esercizi aperti, consente di sviluppare la creatività degli studenti[18]. Inoltre, sfruttando tematiche di interesse per i ragazzi, è possibile aumentare la loro motivazione: un esempio è l'utilizzo di LEGO Mindstorm [29] [13] durante un corso di game design basato sulla Robotica. Gli esercizi aperti risolvono un altro problema: la creatività è spesso associata alla produzione di soluzioni creative a problemi noti, piuttosto che nell'identificazione di nuovi problemi da affrontare; questi ultimi possono essere adoperati per incrementare le capacità di problem-finding, ovvero di individuare nuovi problemi [30].

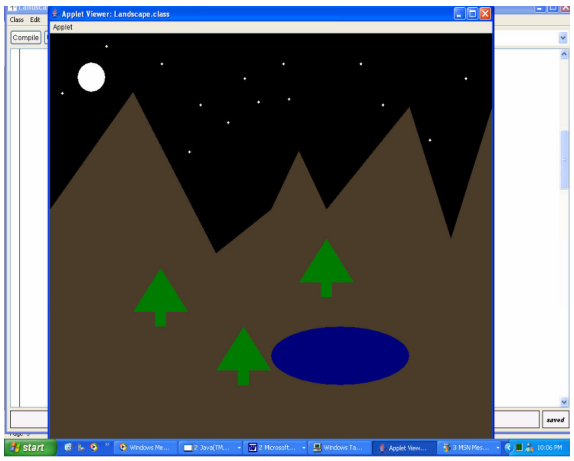

Figura 2.3: Esercizio di programmazione creativa volto alla realizzazione di un paesaggio in Java. Fonte immagine [14]

Un esempio concreto di utilizzo di esercizi aperti durante un corso di introduzione alla programmazione Java in K-12 dimostra come gli studenti, alla fine del semestre, abbiano trovato gli esercizi divertenti e creativi ed abbiano preso dimestichezza con il programma, tanto da condividerne i risultati con familiari ed amici. In figura 2.3 è mostrato un esempio di esercizio aperto, volto alla realizzazione di un paesaggio utilizzando uno lo strumento di programmazione creativa scritto in Java [14].

[13] `www.lego.com/it-it/themes/mindstorms`

Al fine di **ricreare un ambiente creativo**, una delle tecniche utilizzate è quella di mettere in gioco gli studenti, "tirandoli" metaforicamente fuori dalla comfort zone. Questo obiettivo può essere raggiunto in diversi modi:

- realizzando un'aula della creatività, che metta a disposizione strumenti che consentono di stimolare il pensiero creativo, creare un senso di appartenenza e rafforzare i legami del gruppo classe [4]. Esempi sono lavagne multimediali e laboratori;
- mettere a contatto studenti provenienti da diversi contesti scolastici, al fine di generare discussioni interessanti che normalmente non accadrebbero ed inoltre consentire agli studenti che non avrebbero mai considerato l'idea di studiare la programmazione, di riconoscersi come talenti;
- portare gli studenti a diretto contatto con aziende ed industrie, al fine di migliorare le capacità creative ed empatiche.

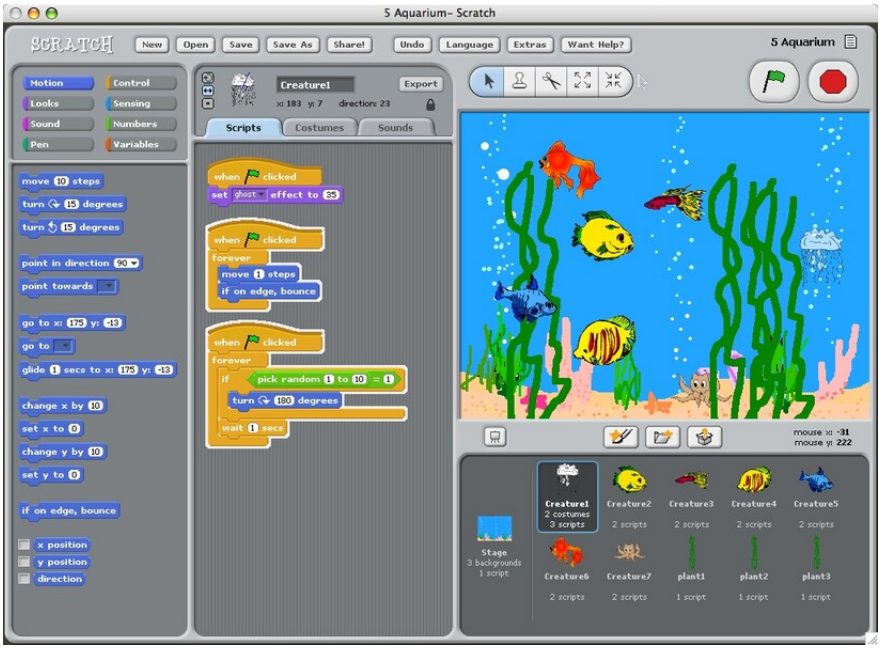

Figura 2.4: IDE Scratch

Un esempio di ambiente creativo lo si trova negli studi condotti in Norvegia, dove un gruppo di ricercatori ha progettato e realizzato due workshop che hanno coinvolto un totale di 29 studenti delle scuole medie. I workshop si basavano sul software open-source Scratch e sull'uso creativo di materiali riciclati all'interno di aule apposite[31]. Nell'immagine 2.4 è possibile vedere un esempio dell'IDE utilizzato durante il workshop.

I risultati raccolti hanno dimostrato che:

- i partecipanti hanno considerato il workshop un'esperienza complessivamente positiva;
- la creatività è un mezzo eccellente per promuovere e insegnare la programmazione;
- l'approccio del workshop suscita interesse per l'informatica, in particolare tra le studentesse.

Questo dimostra, quindi, come l'ambiente in cui gli studenti apprendono gioca un ruolo significativo nel potenziare la creatività degli studenti[3].

Uno strumento per attirare e trattenere più studenti di informatica è quello di **fissare come primo obiettivo la creatività, e la programmazione come secondo**. Questo approccio è dimostrato essere vincente in quanto è in grado di attrarre anche gli studenti che non sono dotati di grandi doti analitiche, che quindi trovano i tradizionali corsi di introduzione alla programmazione tediosi e noiosi. Un esempio è quanto è stato svolto in una scuola superiore Americana, dove due docenti hanno modellato un corso di Creative Computation basato su Processing. Questo corso ha dimostrato come l'integrazione di arti mediali, design e informatica può attrarre e motivare con successo gli studenti ad apprendere i fondamenti della programmazione [1].

Inoltre, consente di superare le barriere di genere e i retaggi culturali, che limitano i giovani dall'inseguire i propri obiettivi. Un esempio è quello di unire il Creative Computation con il Computational Thinking al fine di realizzare dei Computational Creativity Excersizes, che portano ad un aumento dell'apprendimento degli studenti, a prescindere dalla loro motivazione [11].

Un altro esempio è invece il Media Computation [32, 12], un approccio all'informatica adottato nei corsi CS1 che spiega come vengono manipolati i media digitali. In particolare, l'università Georgia Tech [14] richiede che tutti gli studenti, inclusi quelli di Arte, Architettura ed Economia, debbano tenere almeno un corso di informatica. Gli studenti di questi corsi, però, utilizzano il computer e l'informatica come uno strumento di comunicazione, piuttosto che come mezzo per programmare. Mediante questo corso, modificando i pixel di un'immagine o i campioni di un suono per diminuire il volume, hanno imparato a conoscere i loop. Ancora, rimuovendo l'effetto occhi rossi nell'immagine senza modificare gli altri colori o modificando solo parte del suono, hanno appreso il funzionamento delle istruzioni condizionali. Questi sono solo alcuni esempi di questo corso, che lascia completa libertà agli

[14]Georgia Institute of Technology.

studenti riguardo lo strumento da utilizzare: il compito da svolgere è definito in termini di calcolo, ma la scelta di quale supporto utilizzare spetta a loro [32].

2.4 Strumenti esistenti

A seguire, sono presentate le piattaforme e gli strumenti che supportano e promuovono l'uso del Creative Coding, offrendo risorse, ambienti di sviluppo e comunità per gli artisti, designer e programmatori che desiderano esplorare l'intersezione tra arte, design e tecnologia.

2.4.1 Processing

Processing è un ambiente di sviluppo open-source ed un linguaggio di programmazione orientato all'arte e alla creatività, pensato per la costruzione di forme 2D e 3D, utilizzato per creare opere d'arte visiva, installazioni interattive e animazioni. Successore dell'ambiente di programmazione *Design By Numbers* sviluppato da John Maeda [4], Processing si basa su una versione semplificata di Java e un modello di programmazione grafica. Supporta la scrittura del codice in Java, Python e JavaScript [1]: il codice Java può essere inserito liberamente in qualsiasi programma e ogni programma può essere esportato in un Applet o in un'applicazione per SO Linux, macOS e Windows. Ha una curva di apprendimento molto bassa, è semplice, intuitivo e facile da utilizzare. Inoltre, essendo open-source, è possibile estenderlo e modificarlo e può essere eseguito sulle principali piattaforme di programmazione disponibili.

Mette a disposizione un IDE [15] leggero e facile da utilizzare in quanto rimuove le difficoltà che si celano nella programmazione ad oggetti: non è necessario definire una complessa raccolta di classi, piuttosto si utilizzano quelle messe a disposizione. Grazie alla sua facilità d'uso e al fatto che consenta di testare rapidamente piccoli frammenti di codice con il minimo sforzo, risulta un ottimo strumento per i programmatori alle prime armi. Nonostante la facilità d'uso, si tratta di un linguaggio di programmazione robusto e completo, in grado di riprodurre grafica e animazioni complesse, senza dover invidiare nulla ad OpenGL[16] o ad altre librerie grafiche [4].

Processing viene ampiamente adottato dal mondo accademico: dipartimenti di Arte, Design, Architettura e Scienze lo utilizzano per visualizzare componenti grafici; dipartimenti di informatica per insegnare CS1 etc. Numerose sono le

[15]Integrated Development Environment, ovvero ambiente di sviluppo integrato

[16]specifica che definisce una API per più linguaggi e per più piattaforme per scrivere applicazioni che producono computer grafica 3D.

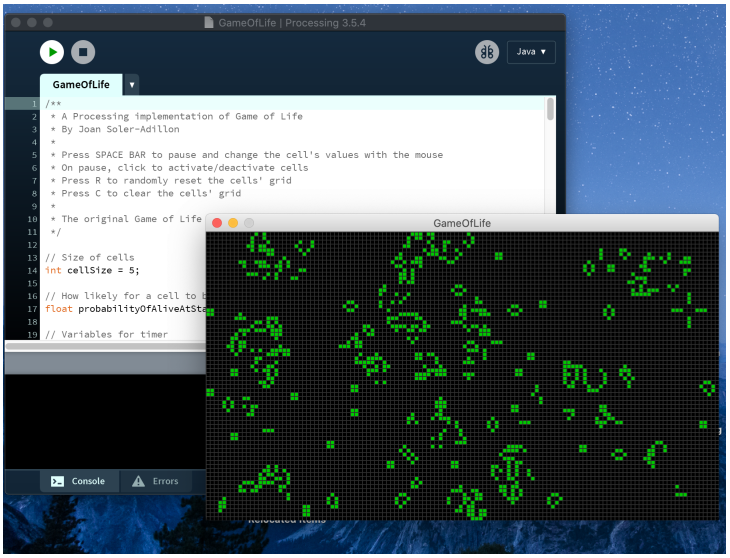

Figura 2.5: Esempio di IDE Processing

testimonianze di utilizzo in corsi di introduzione alla programmazione in classi K-12, CS-0/1/2 [7, 1, 27, 4].

2.4.2 P5.js

P5 è una libreria JavaScript gratuita ed open-source basata su Processing, pensata per facilitare la creazione di arte digitale, visualizzazioni creative e interazioni in ambiente web. Esattamente come Processing, offre un'interfaccia semplice ed intuitiva che consente di creare grafiche e animazioni con le quali l'utente può interagire: a partire da una serie di funzioni predefinite, gli utenti possono realizzare facilmente diversi tipi di forme più o meno complesse. P5 non è pensato solo per gli utenti alle prime armi, infatti grazie alle numerose funzionalità messe a disposizione ogni giorno dalla community, strizza l'occhio anche agli utenti professionisti. In ogni caso, esempi e documentazione sono resi disponibili a tutti, sia ai meno esperti che possono imparare le nozioni di base, sia ai più esperti che possono sbizzarrirsi nell'utilizzo di funzioni più spinte e complesse[33].

Questo strumento non limita alla creazione di semplici disegni o animazioni, ma consente di aggiungere anche testo, suoni, video ed interagire con l'input come la webcam del computer o caricamento di documenti, il tutto grazie ad HTML5.

In particolare, un progetto in p5 è composto da una serie di linguaggi messi insieme:

- Javascript: realizza grafiche interattive mostrate sulla pagina;

- CSS: consente di personalizzare i componenti come input, caricamento file, rendendo il progetto interattivo;

- HTML: collega tutti gli elementi nella pagina.

In questo modo, è possibile realizzare progetti più interattivi, rispetto a Processing che consente di utilizzare solo mouse e tastiera.

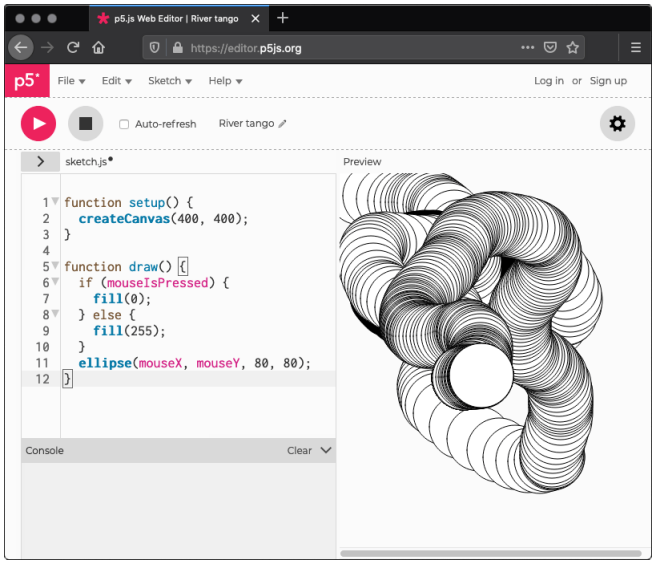

Figura 2.6: Esempio di IDE online p5.js

P5 mette a disposizione un editor online, facilmente accessibile a tutti da diverse piattaforme, ed una libreria che può essere integrata in altri linguaggi ed editor, utilizzata per compilare codice javaScript fornitogli e produrre un componente grafico.

Come si vede nell'immagine 2.6, l'editor online è composto da:

- **editor di testo**: propone un esempio di base da cui partire, include la colorazione della sintassi;

- **console**: consente di visualizzare errori o output;

- **anteprima**: mostra lo sketch una volta che il codice viene compilato ed eseguito. La funzionalità di auto-refresh, consente di generare l'anteprima del progetto senza dover ricompilare.

2.4.3 openFrameworks

openFrameworks [17] è una libreria C++ open-source per la creazione di progetti artistici e interattivi. Basato su una architettura modulare, questo framework risulta facilmente estensibile ed integrabile con altre tecnologie. Molto versatile, è utilizzato per una vasta gamma di applicazioni, dalle installazioni artistiche all'arte digitale [34].

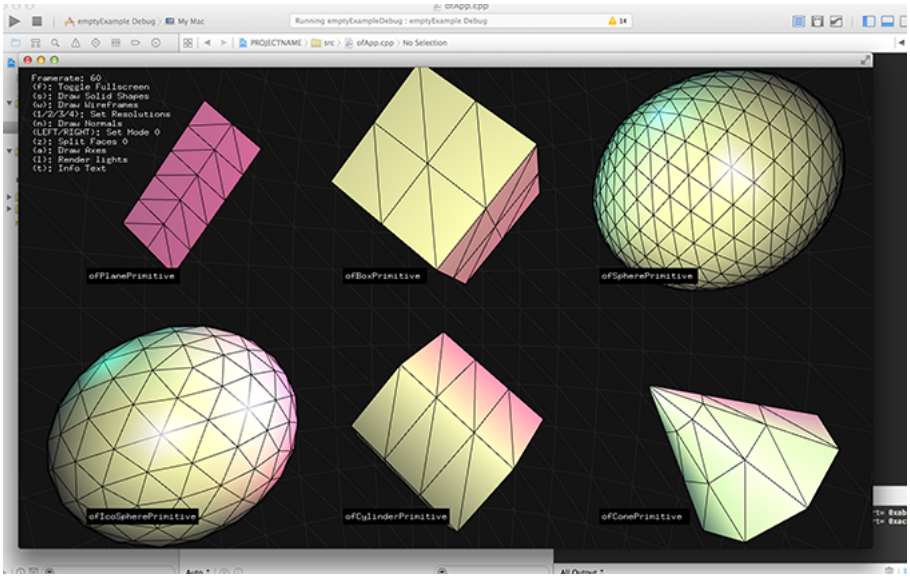

Figura 2.7: Esempio di IDE openFrameworks

Questa piattaforma si impegna a raggiungere una serie di obiettivi:

- **Collaborazione**: diversi utenti contribuiscono ad accrescere l'ecosistema di openFrameworks attraverso discussioni, sviluppo di plugin e progetti. Inoltre, l'approccio principale di questa piattaforma è *"Do It With Others"*: nonostante si promuova la cultura del "fai da te", è importante interagire con altri utenti tramite workshop, conferenze, e community online;
- **Semplicità**: la libreria offre degli esempi modificabili che consentono di accelerare l'apprendimento delle funzionalità di base, anche se questa mette a disposizione funzioni avanzate;
- **Coerenza ed intuitività**: per facilitare l'apprendimento della libreria, il framework cerca di rendere i nomi delle funzioni e delle variabili il più intuitivo possibile;

[17] `openframeworks.cc`

- **Cross-Platform**: può essere eseguita su diversi sistemi operativi come macOS, Windows, Linux, iOS, Android e altri;

- **Estensibilità**: La libreria può essere integrata con librerie più complesse e personalizzate, come addons, plugins, etc.

A differenza degli strumenti analizzati precedentemente, la struttura della libreria può renderlo più complesso da apprendere.

2.4.4 Computational Music

Sebbene il Creative Coding sia spesso associato alle arti grafiche e visive, questo può trovare tante altre forme di espressione, tra cui la musica. Numerosi sono gli strumenti realizzati a tal fine: a seguire verrà presentata una breve carrellata dei principali strumenti e ambienti di programmazione che consentono di esprimere il CC attraverso la musica.

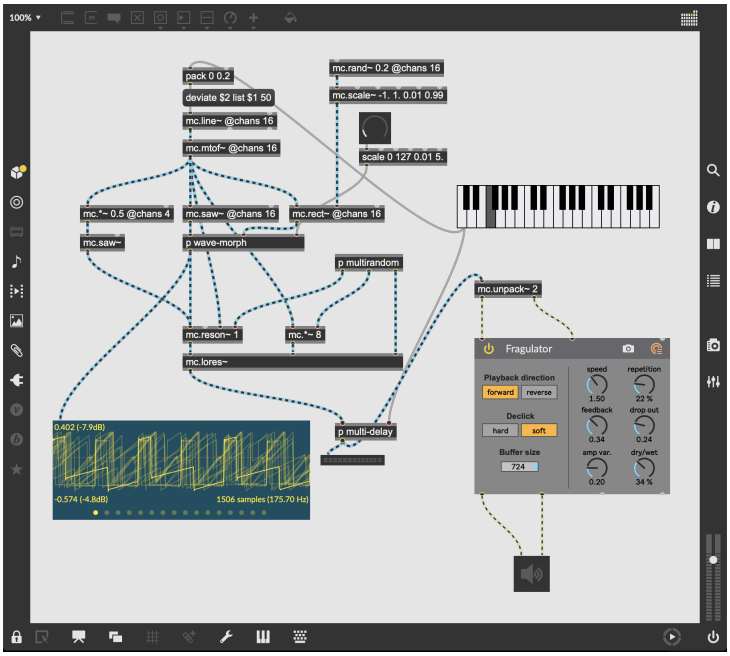

Figura 2.8: Esempio di IDE Max

Max è un ambiente di sviluppo visuale che consente di creare musica e multimedialità in tempo reale. Si basa sull'estensione audio MSP, utilizza Jitter [18]

[18]Motore audiovisivo in tempo reale.

per gestire l'elaborazione delle immagini e dei video. Viene utilizzato per la composizione musicale, la manipolazione di suoni, la creazione di strumenti musicali digitali e performance audiovisive [8].

Le principali caratteristiche di questo framework sono:

- **Programmazione a moduli (patch)**: come si può notare nell'immagine 2.8, Max utilizza un approccio di programmazione visuale in cui gli utenti compongono dei moduli (o patch) e li collegano tra di loro per creare flussi di dati. Questo approccio flessibile consente di manipolare segnali audio, dati e interfacce utente in tempo reale con estrema facilità;
- **UI semplice ed intuitiva**: la sua interfaccia utente intuitiva, permette di creare prototipi e progetti complessi senza scrivere codice.

VVVV [19] è un ambiente di programmazione visuale progettato per la creazione di contenuti multimediali interattivi e per la realizzazione di applicazioni artistiche, progetti di arte interattiva, visualizzazione dati, installazioni e performance.

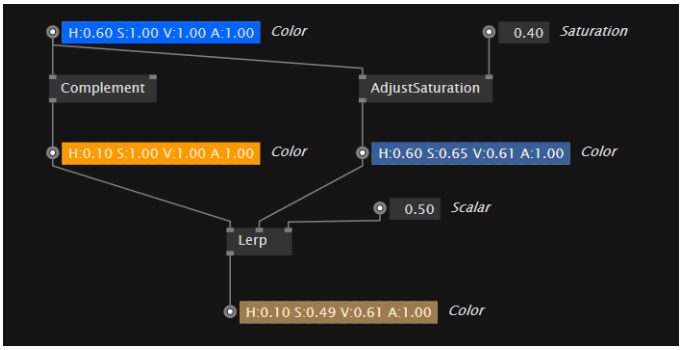

Figura 2.9: Esempio di programmazione basata su nodi in VVVV

Le caratteristiche principali di questo ambiente sono le seguenti:

- **Programmazione basata su nodi**: come mostrato in figura 2.9, gli utenti collegano dei nodi grafici al fine di definire la logica e flussi di dati. È un linguaggio orientato sia all'elaborazione dei media che alla visualizzazione dei dati;
- **Orientato alla grafica 3D e animazioni**: VVVV è uno strumento molto potente nell'elaborazione delle immagini, nella grafica 3D e nella definizione di animazioni, consente infatti di creare scenari e ambienti interattivi.

[19] `visualprogramming.net`

2.5 Risultati di apprendimento

Analizzando i risultati delle attività di potenziamento della creatività nell'apprendimento degli studenti, si evince come l'impatto sia principalmente di tipo **attitudinale**. In particolare, gli strumenti utilizzati e le attività condotte hanno avuto un impatto positivo sugli studenti, che hanno sperimentato:

- **divertimento** [32]
- **motivazione** [2]
- **percezione che l'informatica sia utile in una carriera futura** [4]
- **coinvolgimento con il materiale del corso** [1]
- **intenzione di continuare gli studi informatici** [7]
- **maggiore confidenza nel lavorare in gruppo.**[7]

I primi tre fattori sono stati valutati sulla base dei risultati dei sondaggi condotti, delle interviste e delle valutazioni dei corsi in cui la maggior parte degli studenti ha espresso sentimenti positivi riguardo all'attività svolta.

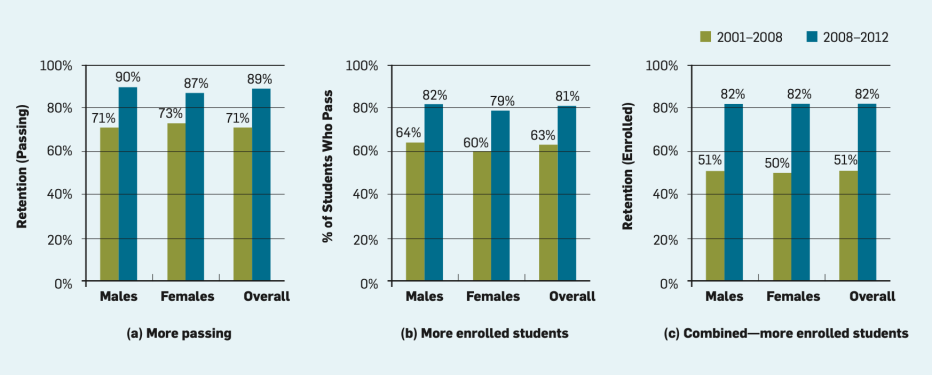

Figura 2.10: Risultati di un corso che combina pair programming (programmazione in coppia), peer-instruction (istruzione tra pari) e media computation in quattro anni. (a) Intenzione di continuare gli studi informatici per gli studenti che superano il corso introduttivo (b) Tassi di successo per gli studenti inizialmente iscritti. (c) Intenzione di continuare gli studi informatici per gli studenti inizialmente immatricolati. Fonte immagine [32]

Inoltre, come si evince nei grafici in 2.10, gli studi condotti sui corsi di Media Computation [32, 12] presentano prove evidenti degli effetti positivi sulla

motivazione e pertinenza sperimentata dagli studenti, in particolar modo per le studentesse.

Questi risultati positivi, però, non si fermano alle sole attitudini degli studenti, ma trovano un riscontro positivo anche nel successo accademico [2]:

- **maggior apprendimento dei concetti del corso** [2]
- **incremento del tempo di studio aggiuntivo per divertimento** [13, 4, 14]
- **diminuzione dell'abbandono scolastico e fallimento dei corsi di informatica** [35].

I corsi di CC offrono un ambiente accessibile e inclusivo che contribuisce a ridurre il divario di genere nel settore tecnologico. Incoraggiando le ragazze a partecipare, tali corsi possono favorire un equilibrio di genere in un campo tradizionalmente dominato dagli uomini. Un esempio lampante è dimostrato dai risultati degli studi condotti in una scuola superiore in America [1], dove due docenti hanno introdotto un corso di programmazione creativa: durante l'anno accademico 2012-2013, anno di prima introduzione del corso, la percentuale di ragazze iscritte era del 16%, rispetto al 52% raggiunto nell'anno accademico 2015-2016. Un altro valido esempio è quanto emerso dai sondaggi condotti all'interno di un college Americano, dove il corso di introduzione alla programmazione introduce il CC e le arti mediante Processing: i risultati sono confrontanti con un'altra scuola che utilizza un approccio tradizionale ed una che utilizza la robotica, e dimostrano come l'approccio creativo sia più apprezzato dagli studenti e studentesse[4].

2.6 Limitazioni

Il Creative Coding è un campo che, sebbene offra opportunità straordinarie, si confronta con diverse limitazioni.

Una delle principali sfide è la **formazione degli insegnanti**: insegnare la creatività agli educatori, affinché questi possano trasmetterla in modo efficace agli studenti, richiede una preparazione specifica. Questo implica un approccio pedagogico che spesso va oltre il semplice insegnamento delle nozioni tecniche, richiedendo risorse aggiuntive, tempo e supporto per acquisire e applicare un metodo di insegnamento creativo. Ciò solleva la questione: come insegnare in modo creativo la creatività agli insegnanti, prima che gli insegnanti possano insegnarla in modo efficace agli studenti?

L'integrazione di attività creative nei corsi introduttivi alla programmazione è un'altra difficoltà. Trovare un equilibrio tra l'offrire libertà di espressione e stabilire confini utili per gli studenti o allineare le attività creative con i contenuti del corso,

richiede un'attenta pianificazione e integrazione curricolare. Questo è fondamentale per garantire un apprendimento efficace e coerente[2]. Inoltre, questi strumenti vengono spesso visti con scetticismo da parte dei docenti e studenti: in alcuni casi si ha la percezione che questi semplifichino troppo il lavoro agli studenti oppure che non siano effettivamente utili al fine dell'apprendimento. Un esempio è quanto riportato in [12], dove alcuni studenti sono preoccupati del fatto che l'auto completamento della sintassi e gli esempi disponibili possano causare la perdita di opportunità di apprendimento.

Infine, la **carenza di strumenti e risorse efficaci per individuare gli errori di programmazione** può costituire un ostacolo significativo. L'ambiente creativo spesso richiede una guida intuitiva per affrontare i problemi, e la mancanza di tali strumenti può rallentare il processo di apprendimento e suscitare frustrazione negli studenti. Questo è quanto sottolineato da Kogan [8], della Università di New York, che durante un corso di introduzione al machine-learning mediante l'utilizzo della computazione creativa, ha evidenziato come la mancanza di astrazione ad alto livello nelle librerie di computazione creativa e la necessità di possedere una esperienza profonda e mirata per un debug efficace ed efficiente siano dei limiti all'apprendimento di questa disciplina.

In definitiva, il Creative Coding offre opportunità significative, ma alcune limitazioni possono presentarsi nella formazione degli insegnanti, nell'integrazione delle attività creative nel curriculum, nell'utilizzo di risorse avanzate e nel supporto disponibile per il debugging e la risoluzione di problemi. Superare queste limitazioni richiede una combinazione di formazione mirata, strumenti intuitivi e risorse di supporto specifiche.

Capitolo 3

Progettazione

Il presente capitolo delinea le scelte progettuali che hanno plasmato Canvas Coding, a partire dall'identificazione dei requisiti emersi durante la fase di analisi, tenendo in considerazione le limitazioni degli strumenti attualmente esistenti basati sull'utilizzo del CC per scopi didattici.

Da questo processo sono emerse tre proposte distinte: un'estensione per un editor di codice, un'applicazione e un gioco. Per ciascuna di esse, sono specificati i requisiti soddisfatti, le limitazioni superate e i vantaggi e svantaggi associati all'implementazione proposta. Tra le tre alternative è stato scelto di sviluppare il gioco e, nel prosieguo del capitolo, ne viene fornita una presentazione dettagliata, accompagnata dalla spiegazione delle ragioni che hanno guidato l'adozione della "gamification". Successivamente, viene esposta la logica alla base del gioco, fondata sul principio della *"direct manipulation"*, che consente agli utenti di interagire direttamente con gli oggetti visualizzati sullo schermo senza dover ricorrere a comandi complessi.

Vengono poi definite le caratteristiche principali e le componenti del gioco, come la suddivisione in livelli e step e la mappatura con i concetti teorici, estratti dal programma didattico del corso di informatica presso un istituto tecnico superiore. Segue la presentazione dei prototipi a bassa e media fedeltà. Infine, il capitolo si conclude con un'esposizione dettagliata delle principali funzionalità del gioco.

3.1 Scelte progettuali

Le decisioni che guidano Canvas Coding prendono in considerazione in primis i bisogni degli studenti, individuati durante la fase di analisi degli articoli a riguardo. In particolare, si pone particolare attenzione ai principali bisogni riscontrati dagli studenti durante i corsi introduttivi alla programmazione, tra i quali:

1. **Visualizzare graficamente i concetti insegnati durante i corsi** [1, 31];

2. **Esprimere la propria creatività** [2, 3, 4];
3. **Essere guidati ma allo stesso tempo essere liberi di esplorare in autonomia** [18, 14];
4. **Motivazione e stimoli, al fine di non perdere interesse** [21, 18, 13, 11];
5. **Collaborare con i propri compagni** [14, 3];
6. **Non essere vincolati a delle indicazioni stringenti** [32];
7. **Condividere le proprie creazioni all'esterno delle mura scolastiche** [31].

Inoltre, ai bisogni degli utenti, si sommano anche le limitazioni delle piattaforme e delle tecniche basate sul CC attualmente disponibili, individuate nel capitolo precedente:

8. **Necessità di formare gli insegnanti all'utilizzo delle piattaforme e metodologie d'insegnamento**;
9. **Integrare le attività creative nei corsi di programmazione** [2];
10. **Carenza di strumenti efficienti per individuare gli errori di programmazione** [8, 36, 30].

3.1.1 Soluzioni proposte

A partire dai bisogni e dalle limitazioni degli strumenti attualmente disponibili, sono state proposte diverse soluzioni che utilizzano il CC a fini educativi. Le più rilevanti sono state le seguenti:

1. **Sviluppo di un'estensione per Visual Studio Code**[1]: realizzare un'estensione che consenta agli studenti di visualizzare graficamente, sotto forma di componenti grafici 3D o immagini, quanto scritto con il codice all'interno dell'editor. Questa soluzione propone un estensione indipendente dal linguaggio di programmazione che consente agli utenti di creare in modo creativo immagini, forme ed oggetti complessi mediante l'utilizzo del codice. In particolare, una volta definito il codice necessario, l'estensione legge quanto scritto dall'utente e produce su una schermata apposita la rappresentazione grafica associata. In caso di errore, un messaggio contenente i dettagli viene mostrato;

[1] editor di codice sorgente sviluppato da Microsoft.

2. **Realizzazione di un'applicazione Android**: sviluppare un'applicazione che consenta di creare elementi grafici e generare il codice corrispondente. A partire da una tela bianca oppure utilizzando degli sfondi pre-caricati, l'applicazione consente agli utenti di definire ed inserire elementi all'interno del paesaggio e di modificarne le caratteristiche. Una volta terminato, l'applicazione consente di condividere le proprie creazioni ed esportare il codice generato in un determinato linguaggio, da testare successivamente sul proprio IDE preferito;

3. **Realizzazione di un gioco online**: sviluppare un gioco online, suddiviso in livelli ed obiettivi, che consente di scrivere del codice e visualizzare i componenti grafici generati da questo e viceversa. Inoltre, associare ad ogni livello uno o più argomenti trattati durante le lezioni teoriche, per consentire agli studenti di proseguire senza supervisione del docente ed allo stesso modo ripassare i concetti in modo innovativo.

Nella tabella 3.1 sono indicati, per ogni proposta, quali bisogni degli studenti consente di soddisfare e quali limitazioni risolve. Da notare come la riga relativa alla formazione dei docenti sia in logica negata.

Obiettivi e limitazioni	**Estensione**	**Applicazione**	**Gioco online**
1 Visualizzare concetti	SI	SI	SI
2 Creatività	SI	SI	SI
3 Indicazioni e guida	NO	SI	SI
4 Motivazione	SI	SI	SI
5 Collaborazione	NO	SI	SI
6 Compiti aperti	SI	SI	SI
7 Condivisione	SI	NO	SI
8 NO formazione	NO	SI	SI
9 Integrazione nei corsi	SI	SI	SI
10 Individuare errori	NO	NO	SI

Tabella 3.1: Confronto delle soluzioni individuate

3.1.2 Pro e contro

Per ciascuna delle soluzioni proposte, sono delineati i vantaggi e gli svantaggi associati al loro eventuale impiego, confrontati con le alternative già utilizzate emerse nella fase di analisi.

Per quanto riguarda lo **Sviluppo di un'estensione per VS Code**, i principali vantaggi sono:

1. **Strumento innovativo**: confrontando con gli studi condotti, nessuna estensione che utilizza il CC a fini didattici sembra essere mai stata sviluppata.

2. **Riusabilità**: se sviluppata per essere indipendente dal linguaggio utilizzato, questa soluzione risulta molto adattabile a vari contesti educativi dove spesso si utilizzano linguaggi differenti in base alla scuola scelta. La maggior parte delle soluzioni trovate in [7, 3, 1, 35, 4, 12] sono strettamente legate ad un singolo linguaggio di programmazione o strumento.

Le limitazioni individuate durante la fase di analisi sono le seguenti:

1. formare gli insegnanti all'utilizzo dello strumento ed i tecnici di laboratorio all'installazione e configurazione sui computer scolastici;

2. necessita della guida da parte di un docente che indichi quali azioni svolgere, sotto forma di compiti per casa o in classe;

3. scetticismo da parte dei docenti che potrebbero portare gli studenti a non utilizzare lo strumento. Come indicato in [12], "l'utilità percepita è una parte centrale del fatto che un sistema venga utilizzato"; siccome questo strumento necessita della guida da parte del docente, se questa viene a mancare a causa dello scetticismo, anche l'utilità dello strumento potrebbe venir meno;

4. difficoltà nell'utilizzo all'esterno delle mura scolastiche, in quanto necessita dell'installazione sui computer personali degli studenti.

I vantaggi individuati nella **realizzazione di un'applicazione Android** sono i seguenti:

1. **Nessuna guida necessaria**: a differenza delle soluzioni che utilizzano Processing [7, 1, 35, 4], P5.js [22, 12] o altri strumenti come Scratch [21, 1, 4], VVVV, Max [22], l'applicazione è guidata e non necessita di particolari introduzioni da parte del docente.

2. **Tecnologia famigliare per gli studenti**: utilizzare una piattaforma alla quale questi sono abituati, potrebbe risultare uno strumento efficace ai fini di alleviare l'impatto sperimentato dagli studenti novizi quando utilizzano l'ambiente di programmazione per la prima volta. Benché mai utilizzata precedentemente, un'applicazione presenta un impatto meno intimidatorio rispetto a uno strumento completamente sconosciuto [13, 36], come ad esempio un ambiente di sviluppo integrato (IDE).

Anche in questo caso, diverse insidie e difficoltà sono state individuate:

1. necessario utilizzare i tablet di proprietà degli studenti in quanto non fanno parte della dotazione scolastica, a differenza dei computer disponibili nelle aule multimediali;

2. l'applicazione è vincolata alla piattaforma per la quale è stata sviluppata, ma gli studenti potrebbero avere dispositivi non compatibili;

3. difficoltà nell'integrazione con i concetti teorici insegnati da parte dei docenti;

4. non consente di individuare gli errori di programmazione, in quanto demandato all'IDE su cui lo studente decide di eseguire il codice.

Infine, i vantaggi individuati nella **realizzazione di un gioco online** sono i seguenti:

1. **Utilizzo del gamification per motivare gli studenti**: questo approccio si è dimostrato efficace nel migliorare la motivazione degli studenti durante i corsi di introduzione alla programmazione [21];

2. **Indipendenza dalla piattaforma**: può essere eseguito liberamente su computer e tablet basati su diversi SO;

3. **Facilità d'integrazione con i concetti teorici**: ogni livello ed ogni obiettivo che lo compone è incentrato su una particolare tematica insegnata durante il corso. Per questo motivo, è molto semplice utilizzarlo come strumento da accompagnare alle lezioni teoriche o per ripassare i concetti;

4. **Nessuna guida da parte del docente**: è il gioco stesso che, mediante le richieste di ogni step, guida lo studente attraverso le nozioni teoriche;

5. **Implementazione meno onerosa**: grazie alle numerose librerie grafiche disponibili compatibili con i principali framework, lo sviluppo risulta meno complesso e più estensibile e manutenibile in futuro.

Gli svantaggi individuati sono i seguenti:

1. Supporto limitato di linguaggi: basandosi sui concetti del paradigma di programmazione ad oggetti, si adatta con difficoltà a linguaggi che non utilizzano questo paradigma;

2. Necessita di conoscenze di base dei concetti, può essere utilizzato per rafforzare le nozioni teoriche o per ripassare determinati esercizi.

3.1.3 Soluzione scelta

Dopo un'analisi approfondita delle tre proposte, considerando attentamente i loro vantaggi e svantaggi, la decisione è stata quella di procedere con lo sviluppo di un gioco online. Le ragioni che hanno guidato questa scelta, oltre ai vantaggi evidenziati nella sezione precedente, sono le seguenti:

1. **Fusione del Gamification con Creative Coding**: La gamification consiste nell'applicazione di elementi tipici dei giochi in contesti non ludici al fine di migliorare l'interazione, la motivazione e il comportamento degli utenti. L'obiettivo è quello di coinvolgere le persone e motivarle ad adottare determinati comportamenti, apprendere nuove abilità o partecipare attivamente a determinate attività. Gli elementi tipici di questo approccio includono punti, classifiche, premi, livelli, missioni o sfide. Numerosi strumenti esistenti che utilizzano il gamification sono stati individuati nella fase di analisi, come Protobject [21], Greenfoot [9], Blockly [23]. Tutti questi strumenti, però, hanno in comune la caratteristica di non utilizzare direttamente il Creative Coding ma di focalizzarsi sull'acquisizione del pensiero computazionale, mediante tecniche disparate come Computer Vision per Protobject ed altre. Canvas Coding, invece, propone l'utilizzo del gamification per rinforzare la sfera motivazionale degli studenti e il Creative Coding per divertire e coinvolgere gli alunni al fine di raggiungere risultati di apprendimento migliori.

2. **Raggiunge gli obiettivi e supera le limitazioni**: come si vede in tabella 3.1, delle tre proposte, il gioco online è l'unico che consente di soddisfare tutti gli obiettivi stabiliti.

3.1.4 Logica di base

Al fine di rendere l'interazione uomo-computer più intuitiva, naturale e immediata, Canvas Coding si basa sul paradigma chiamato *Direct Manipulation*: definito e promosso inizialmente da Shneiderman nel 1982 [37], consente agli utenti di interagire direttamente con gli oggetti visibili sullo schermo o con gli elementi dell'interfaccia, manipolandoli o agendo su di essi in modo diretto [9]. Questo approccio mira a rendere più tangibili le azioni dell'utente, riducendo la distanza tra l'intenzione dell'utente e l'effetto prodotto sull'interfaccia. Esempi di questo approccio sono: PowerPoint, Excel, Word, Adobe Illustrator.

Approccio opposto è il *Programmatic Manipulation*, o manipolazione programmatica, si riferisce alla manipolazione di sistemi e processi attraverso l'uso di programmazione o codice informatico. Questo concetto può essere applicato a diversi campi, tra cui informatica, economia, politica etc.. Esempi di programmatic manipulation sono Latex ed SQL.

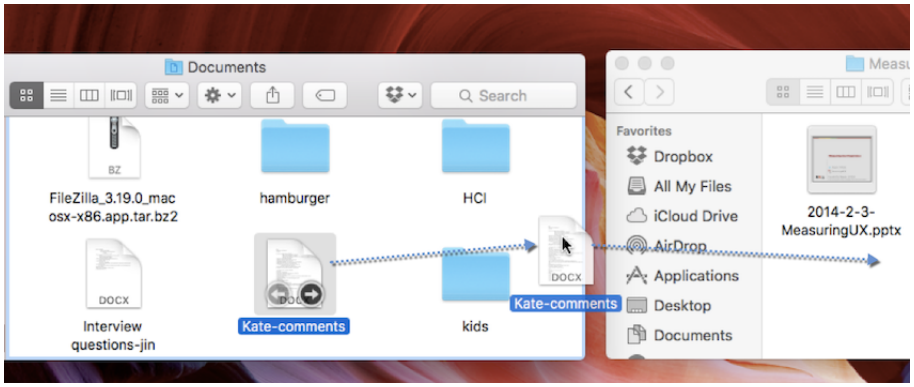

Figura 3.1: Direct Manipulation in HCI

I principi fondamentali dell'approccio Direct Manipulation includono:

- **Manipolazione diretta degli oggetti**: Gli utenti possono spostare, modificare, e manipolare oggetti o elementi dell'interfaccia direttamente sullo schermo senza dover ricorrere a comandi scritti.
- **Feedback immediato**: viene generato un feedback immediato a partire dalle azioni svolte sull'interfaccia. Ad esempio, trascinando un oggetto, si vedrà il suo movimento in tempo reale.
- **Azioni visibili e reversibili**: Le azioni intraprese dagli utenti sono visibili e reversibili. Ad esempio, è possibile annullare o ripetere un'azione facilmente e in modo comprensibile.
- **Rappresentazione del mondo reale**: L'interfaccia mira a riflettere il mondo reale, consentendo agli utenti di interagire con gli oggetti digitali in modo simile a come interagirebbero con gli oggetti fisici. Esempi sono il drag-and-drop [2], pinch-to-zoom [3].

L'obiettivo principale di questo approccio è quello di migliorare l'usabilità e l'esperienza utente, riducendo la curva di apprendimento e consentendo interazioni più intuitive con i sistemi informatici. Questo concetto ha influenzato lo sviluppo di molte interfacce utente, come ad esempio le interfacce touch screen, in cui gli utenti possono toccare direttamente gli elementi sullo schermo per interagirci, riducendo la necessità di comandi intermedi.

[2]indica una successione di tre azioni: click su un oggetto, trascinamento in un'altra posizione, rilascio.

[3]azione che consiste nel pizzicare lo schermo per ingrandire.

3.2 Architettura generale

In questa sezione, viene delineata la struttura della piattaforma di gioco Canvas Coding, indicando i componenti da implementare, la loro disposizione nell'interfaccia e le interazioni che si verificano tra l'utente e il gioco. Sfruttando i risultati di questa fase, si procede nella sezione successiva alla creazione dei prototipi a bassa e media fedeltà. Successivamente, è presentato un estratto del programma didattico di informatica degli anni 3 e 4 presso un istituto tecnico superiore italiano, suddiviso in UDA [4] ed attività. A partire da questa suddivisione in concetti teorici, si definisce la struttura del gioco, dove ciascun livello, correlato a una o più UDA, è suddiviso in ulteriori step o obiettivi. Per concludere, sono presentate le principali funzionalità previste dal gioco: per ognuna è indicato il componente logico al quale fa riferimento e le interazioni con l'utente.

3.2.1 Struttura

La struttura del gioco Canvas Coding è stata concepita tenendo in considerazione l'architettura di un IDE tradizionale e il concetto di Direct Manipulation. Al fine di garantire una familiarità anche per coloro aventi esperienza in programmazione tradizionale, devono essere inclusi gli elementi caratteristici di un IDE, quali: un editor di testo, una console di output e un explorer dei file del progetto. Questi tre componenti risultano indispensabili per rendere l'ambiente di sviluppo riconoscibile e intuitivo.

Per quanto riguarda il paradigma del Direct Manipulation, la sua natura innovativa richiede una progettazione creativa e l'introduzione di soluzioni originali. Partendo dai principi generali di questo paradigma, è possibile identificare alcune componenti chiave da integrare: la prima dovrebbe consentire una manipolazione diretta degli oggetti, assumendo la forma di una "finestra" nel mondo 3D; la seconda, invece, deve facilitare l'accesso agli strumenti e alle opzioni disponibili nel gioco, posizionandosi in modo strategico vicino alla precedente per concentrare l'area di azione della sezione grafica in una zona coesa.

Inoltre, è essenziale prevedere un componente dedicato alla comunicazione diretta tra il gioco e l'utente, posizionato in modo prominente e sempre in primo piano.

Partendo da queste specifiche, sono stati identificati i seguenti componenti logici da inserire e integrare nel contesto del gioco:

- **Canvas:** componente grafico fondamentale dedicato alla visualizzazione, modifica e interazione con oggetti 3D e animazioni. Posizionato strategicamente accanto al codice, svolge un ruolo cruciale nell'agevolare l'apprendimento dei

[4] Unità di apprendimento.

concetti teorici fornendo un esempio immediato e tangibile di quanto scritto dall'utente. Questo componente non solo permette la visualizzazione diretta del codice, ma facilita anche l'interazione bidirezionale: da un lato, l'utente può modificare gli oggetti e le animazioni 3D utilizzando l'interfaccia grafica, e, dall'altro, è in grado di osservare le modifiche effettuate tramite questa interfaccia riflesse direttamente nel codice, fornendo così un feedback visivo e operativo completo;

- **Codice:** cosi come in tutti gli ambienti di sviluppo, questo componente svolge il ruolo di un editor di testo, potenziato però con tutti gli strumenti necessari ad un programmatore come l'evidenziazione della sintassi, auto-completamento del codice, sottolineatura nel caso di errore. Inoltre, è prevista la funzionalità di evidenziazione della riga in base al componente grafico selezionato o modificato nel Canvas: in questo modo, l'utente saprà subito riconoscere nel codice quali sono le conseguenze delle azioni svolte mediante l'interfaccia grafica;

- **Side Menu:** posto in prossimità del Canvas, rappresenta una risorsa centrale ed offre all'utente un accesso agevole a tutti gli strumenti necessari per operare efficacemente nella sezione grafica. Un esempio chiaro è il fornire un elenco completo degli oggetti disponibili per l'inserimento, qualora l'utente desideri aggiungerne uno al Canvas. Tuttavia, la sua utilità non si limita solo a questa funzionalità: il Menù Grafico offre anche un accesso immediato ad altre opzioni e funzionalità pertinenti all'interfaccia grafica, quali settaggi, preferenze e scorciatoie. La sua collocazione non solo facilita l'operatività dell'utente ma mira anche a ottimizzare l'esperienza complessiva, fornendo un punto centrale per l'esplorazione e l'impiego delle risorse grafiche disponibili;

- **Action Menu:** un elemento chiave del sistema è il componente dedicato alla comunicazione diretta tra il gioco e l'utente, il quale ha il compito di fornire istruzioni riguardanti le azioni necessarie per completare gli step del livello. La sua particolarità consiste nel mantenere una presenza costante in primo piano, assicurando che le indicazioni siano sempre visibili senza necessità di essere richiuse o cercate attraverso interazioni aggiuntive. Questo design mira a garantire una chiara e immediata consultazione delle azioni richieste, contribuendo così a una user experience fluida e intuitiva. La sua posizione prominente nell'interfaccia sottolinea l'importanza di questo componente nel guidare l'utente attraverso il processo di gioco in modo efficace e senza ostacoli.

- **Console:** componente presente in tutti gli IDE, mostra l'output della computazione e gli errori generati in fase di compilazione;

- **Explorer:** per mantenere un richiamo agli ambienti di sviluppo tradizionali, questo componente consente agli utenti di muoversi nei file e nei progetti recenti.

Una volta identificati i componenti chiave da inserire nel gioco, è stata pianificato il loro posizionamento all'interno della pagina. Tale disposizione segue un approccio che rispetta la distinzione tra i componenti tradizionalmente associati agli IDE (Codice, Console e Browser) da un lato, e quelli connessi alla rappresentazione visiva (Canvas, Side Menu) dall'altro. Questa scelta è guidata dal principio di agevolare il flusso di interazione tra le due sezioni, cercando di rendere la navigazione tra gli elementi del gioco più intuitiva e fluida. In questo modo, si mira a creare un ambiente armonioso e organico che permetta agli utenti di interagire in modo efficiente e coerente con le diverse funzionalità offerte, senza doverle cercare in modo confuso nella pagina. In figura 3.2 segue un esempio della disposizione citata.

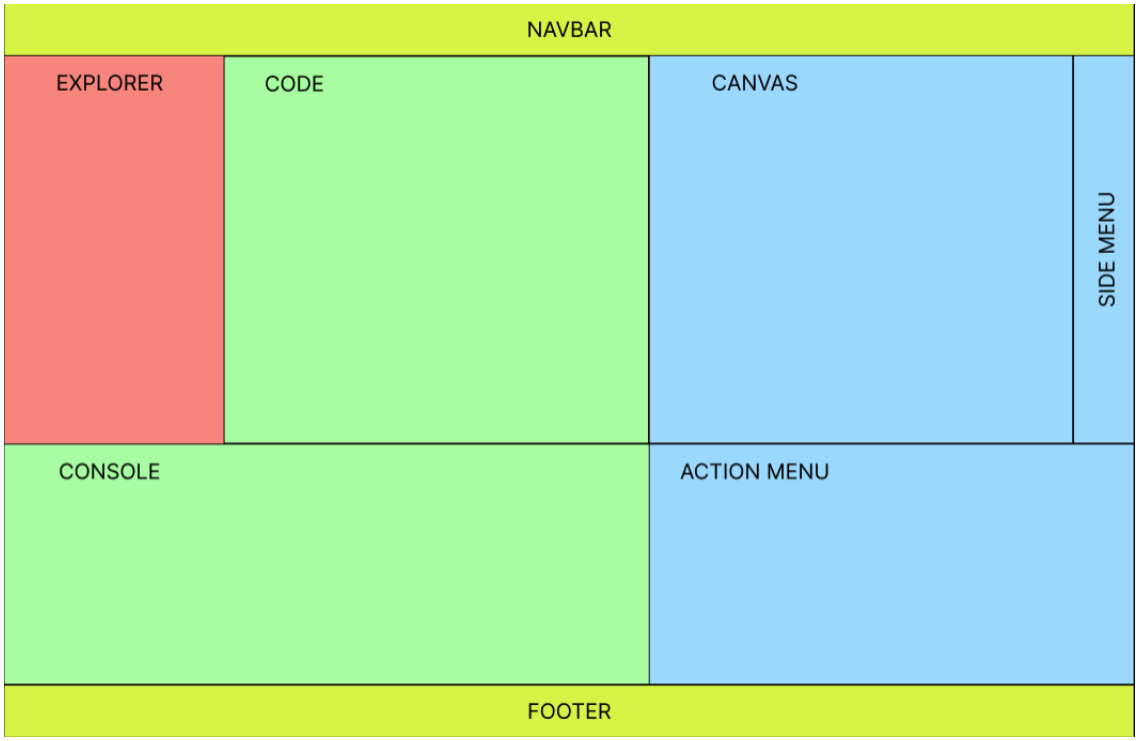

Figura 3.2: Posizionamento dei componenti nel gioco

3.2.2 Argomenti teorici

Attraverso il coinvolgimento dei docenti di informatica e di laboratorio di un istituto tecnico italiano, è stato reso possibile l'accesso a un estratto dettagliato

del programma scolastico impartito durante il terzo e quarto anno.

Tale estratto è accuratamente presentato nella tabella 3.2 in cui sono evidenziate, per ogni colonna, le informazioni cruciali per comprendere la struttura e gli obiettivi del curriculum educativo.

Anno	**UDA**	**Argomenti e attività**
3	Gli algoritmi (UDA1)	Concetto di algoritmo e caratteristiche
3	Elementi di base del linguaggio Java (UDA1)	• Struttura dei programmi, sintassi e semantica del linguaggio • Classe e librerie in Java • Tipi di dati ed operatori aritmetici e logici • Standard input ed output • Interpretazione e compilazione dei programmi
3	Costrutti selettivi ed iterativi del linguaggio (UDA1)	Strutture di controllo (sequenza, selezione, ripetizione)
3	La struttura di dati array (UDA2)	• Struttura dati array (monodimensionali e matrici) • Strutture di controllo per la ricerca dei dati all'interno degli array • Algoritmi di ordinamento
3	I sottoprogrammi (UDA2)	• Funzioni e procedure all'interno della programmazione (metodi Java) • Differenza tra parametri formali ed attuali • Logica ricorsiva

3	Gli oggetti e le classi (UDA3)	• Concetti di oggetto, classe ed istanza e differenze • Dichiarazione ed utilizzo di una classe in Java • Variabili istanza • Metodi • Comandi Java per la creazione di oggetti • Incapsulamento • Interfaccia di una classe
3	L'analisi ed il diagramma delle classi (UDA3)	• Diagramma delle classi • Associazioni tra classi
3	Mascheramento dell'informazione (UDA3)	• Definizione del concetto di Information hiding • Metodi di istanza ed il metodo toString
3	Vettori come attributi di una classe (UDA4)	Attributi strutturati di una classe: i vettori
4	Array di oggetti (UDA1)	Tecnica di gestione degli array di oggetti
4	Proprietà fondamentali della oop (UDA1)	1. Ereditarietà e polimorfismo 2. Gerarchie di oggetti

Tabella 3.2: UDA ed argomenti estratti dal programma di informatica di un istituto tecnico superiore

3.2.3 Livelli e step

A partire dagli argomenti in tabella 3.2, che ricoprono tutte le UDA del 3 anno e la prima del 4 anno, sono definiti i **livelli** che compongono il gioco: ogni livello è associato ad una o più UDA, e tratta diversi argomenti teorici.

Per passare al livello successivo, all'utente è chiesto di completare uno o più step mediante l'utilizzo del codice, Canvas, strumenti nel Side Menu o altre funzionalità presenti nel gioco (ad esempio, tornare al livello corrente oppure compilare il codice). Solo una volta completati tutti gli step di un livello, è possibile passare al successivo.

A seguire, sono presentati i livelli progettati, ognuno associato agli argomenti teorici trattati, con i relativi **step**. Per ogni step, è indicato in tabella:

- Nome step
- Richiesta del gioco
- Risposta dell'utente
- Cambiamento nel gioco.

Il **livello 1** consiste nella definizione di un oggetto grafico 3D mediante l'utilizzo del codice e dell'interfaccia. L'utente aggiunge l'oggetto richiesto e ne personalizza le caratteristiche utilizzando il codice e gli strumenti disponibili. Gli step associati a questo livello sono riportati in tabella 3.3.

Nome	**Richiesta**	**Risposta**	**Cambiamento**
Selezione di un oggetto da aggiungere nel Canvas	In Action Menu, chiede di selezionare uno degli oggetti disponibili nel Side Menu	L'utente seleziona dal Side Menu l'oggetto che vuole aggiungere	L'Action Menu viene aggiornato con le indicazioni successive
Aggiunta dell'oggetto selezionato nel Canvas	In Action Menu, chiede di inserire l'oggetto selezionato all'interno del Canvas	L'utente seleziona seleziona una casella nel Canvas	Il main viene popolato con l'istanza dell'oggetto aggiunto e con la classe; Nel Canvas viene mostrato l'oggetto inserito con le caratteristiche di default

Personalizzazione dell'oggetto	In Action Menu, chiede di personalizzare l'oggetto inserito mediante il codice	L'utente definisce le caratteristiche dell'oggetto mediante metodi set oppure modificando i parametri del costruttore; L'utente compila il codice.	Viene aggiornato il Canvas con le informazioni inserite; L'esito della compilazione viene mostrato nella console di debug.

Tabella 3.3: Step del livello 1

Questo livello copre i seguenti argomenti:

- Tipi di dati (3 Anno, UDA 1)
- Struttura dei programmi, sintassi e semantica del linguaggio (3 Anno, UDA 1)
- Classe e librerie in Java (3 Anno, UDA 1)
- Interpretazione e compilazione dei programmi (3 Anno, UDA 1)
- Metodi (3 Anno, UDA 2)
- Comandi Java per la creazione di oggetti (3 Anno, UDA 3)
- Variabili istanza (3 Anno, UDA 3).

Il **livello 2** consiste nella definizione e nell'ordinamento grafico degli oggetti creati, mediante l'utilizzo del codice e delle funzionalità disponibili. L'utente modifica le caratteristiche ordinabili degli oggetti definiti, come ad esempio le dimensioni per poter ordinare in base al volume. Successivamente, grazie alle funzionalità disponibili nel menu di ordinamento, può generare il codice necessario, variando algoritmi e parametri.

Per ogni ordinamento lanciato, viene generato il rispettivo codice e gli oggetti vengono disposti ordinatamente nel Canvas a seconda del tipo di ordinamento richiesto. L'utente può quindi esportare il codice generato, utilizzando la funzione apposita.

Gli step associati a questo livello sono riportati in tabella 3.4.

Nome	**Richiesta**	**Risposta**	**Cambiamento**

Aggiunta di altri oggetti nel Canvas	In Action Menu, chiede all'utente di creare ulteriori varianti dell'oggetto	L'utente seleziona l'oggetto dal Side Menu e lo inserisce nel Canvas. Ripete questa procedura almeno 2 volte	Viene aggiornato il Canvas con le informazioni inserite; Il main viene popolato con l'istanza dell'oggetto aggiunto e con la classe se il tipo di oggetto inserito non era già presente; L'esito della compilazione viene mostrato nella console di debug.
Modifica delle caratteristiche degli oggetti inseriti	In Action Menu, chiede all'utente di modificare degli attributi ordinabili come le dimensioni degli oggetti aggiunti	L'utente modifica le caratteristiche degli oggetti mediante metodi set o costruttore e compila il codice	Viene aggiornato il Canvas con le informazioni inserite; L'esito della compilazione viene mostrato nella console di debug
Ordinamento degli oggetti attraverso opzione nel Side Menu	In Action Menu, chiede all'utente di accedere all'opzione Sort dal Side Menu	L'utente seleziona Sort dal Side Menu e setta le opzioni di ordinamento richieste (ordine, attributo di ordinamento)	Viene generato l'algoritmo di ordinamento di default ed inserito nel codice; Gli oggetti vengono disposti in modo ordinato nel Canvas; L'esito della compilazione viene mostrato nella console di debug
Ordinamento degli oggetti attraverso opzione nel Side Menu con algoritmi differenti	In Action Menu, chiede all'utente di accedere all'opzione Sort dal Side Menu e di selezionare un algoritmo differente	L'utente seleziona Sort dal Side Menu e setta le opzioni di ordinamento richieste (ordine, attributo di ordinamento, algoritmo)	Viene generato l'algoritmo di ordinamento selezionato ed inserito nel codice; Gli oggetti vengono disposti in modo ordinato nel Canvas; L'esito della compilazione viene mostrato nella console di debug

Tabella 3.4: Step del livello 2

Questo livello copre i seguenti argomenti:

- Strutture di controllo: sequenza (3 Anno, UDA 1)
- Concetto di algoritmo e caratteristiche (3 Anno, UDA 1)
- Concetti di oggetto, classe, istanza e differenze (3 Anno, UDA 2)
- Struttura dati array (3 Anno, UDA 2)
- Strutture di controllo per la ricerca dei dati all'interno degli array (3 Anno, UDA 2)
- Algoritmi di ordinamento (3 Anno, UDA 2)

Nel **livello 3**, l'utente è chiamato a risolvere dei percorsi intricati visualizzati nel Canvas, utilizzando sia il codice che le funzionalità integrate nel gioco. L'obiettivo consiste nella risoluzione di percorsi e nella capacità di evitare gli ostacoli presenti nel Canvas, suddiviso in una sorta di scacchiera. Un elemento aggiuntivo è rappresentato dalla presenza di un avatar, incarnato da un simpatico coniglio, che deve seguire il percorso tracciato dall'utente per raggiungere una specifica destinazione, superando gli ostacoli tatticamente inseriti lungo il tragitto. L'utente ha la possibilità di interagire con il gioco sia tramite il click del mouse sul Canvas che attraverso la scrittura di codice, gestendo in modo dinamico e coinvolgente il movimento dell'avatar fino al raggiungimento della meta indicata. Gli step associati a questo livello sono riportati in tabella 3.5.

Nome	**Richiesta**	**Risposta**	**Cambiamento**
Risoluzione del percorso senza ostacoli	In Action Menu, viene chiesto all'utente di muovere l'avatar al fine di raggiungere la destinazione evidenziata	Mediante il cursore, l'utente seleziona i riquadri disponibili nel Canvas	Viene aggiornato il Canvas mostrando i nuovi riquadri di movimento disponibili; Viene inserita nel codice l'istruzione di movimento contenente le coordinate del riquadro selezionato;

Movimento dell'avatar	In Action Menu, chiede all'utente di completare il percorso	Mediante il cursore, l'utente seleziona i riquadri disponibili nel Canvas sino al raggiungimento della destinazione evidenziata	Viene aggiornato il Canvas mostrando i nuovi riquadri di movimento disponibili; Una volta raggiunta la destinazione, un'animazione mostra il movimento dell'avatar attraverso il percorso tracciato, sino al raggiungimento della destinazione Viene inserita nel codice l'istruzione di movimento contenente le coordinate del riquadro selezionato;
Risoluzione del percorso con ostacoli	In Action Menu, viene chiesto all'utente di muovere l'avatar al fine di raggiungere la destinazione evidenziata, evitando gli ostacoli nel percorso	Mediante il cursore ed il codice, l'utente seleziona i riquadri disponibili nel Canvas	Viene aggiornato il Canvas mostrando i nuovi riquadri di movimento disponibili; Viene inserita nel codice l'istruzione di movimento contenente le coordinate del riquadro selezionato; Una volta raggiunta la destinazione, viene mostrata l'animazione di movimento dell'avatar nel Canvas.

Tabella 3.5: Step del livello 3

Questo livello copre i seguenti argomenti:

- Strutture di controllo: sequenza, selezione, ripetizione (3 Anno, UDA 1)
- Interpretazione e compilazione dei programmi (3 Anno, UDA 1)
- Concetto di algoritmo e caratteristiche (3 Anno, UDA 1)
- Operatori aritmetici e logici (3 Anno, UDA 1)

- Funzioni e procedure all'interno della programmazione (3 Anno, UDA 2)
- Vettori come attributi strutturati di una classe (3 Anno, UDA 4)

Il **livello 4** coinvolge l'utente nella realizzazione dettagliata di un avatar personalizzato, rappresentante il proprio profilo all'interno del gioco. Questa attività richiede l'utilizzo sinergico del codice e dell'interfaccia del gioco. L'utente è chiamato a plasmare il proprio avatar definendone non solo gli elementi basilari come nome, cognome, età, ma anche aspetti più creativi e personalizzati, quali il colore dei capelli, degli occhi, l'altezza e altri dettagli che rendono unico il personaggio. Questa esperienza di creazione consente agli utenti di esprimere la propria individualità, integrando aspetti pratici e creativi. Gli step associati a questo livello sono riportati in tabella 3.6.

Nome	**Richiesta**	**Risposta**	**Cambiamento**
Definizione delle caratteristiche del profilo personale	In Action Menu, chiede all'utente di usare il costruttore per definire le caratteristiche del profilo personale	L'utente popola il costruttore con le informazioni richieste e compila il codice	Viene aggiornato il Canvas con le informazioni inserite; L'esito della compilazione viene mostrato nella console di debug
Definizione delle caratteristiche secondarie	In Action Menu, chiede all'utente di definire caratteristiche secondarie: colore capelli, occhi..	L'utente definisce le caratteristiche secondarie mediante metodi set e compila il codice	Viene aggiornato il Canvas con le informazioni inserite; L'esito della compilazione viene mostrato nella console di debug

Tabella 3.6: Step del livello 4

Questo livello copre i seguenti argomenti:

- Tipi di dati (3 Anno, UDA 1)
- Struttura dei programmi, sintassi e semantica del linguaggio (3 Anno, UDA 1)
- Classe e librerie in Java (3 Anno, UDA 1)
- Interpretazione e compilazione dei programmi (3 Anno, UDA 1)
- Metodi (3 Anno, UDA 2)
- Comandi Java per la creazione di oggetti (3 Anno, UDA 3)

- Variabili istanza (3 Anno, UDA 3).

Il **livello 5** apre all'utente l'esplorazione di un mondo condiviso, un ambiente virtuale dove è possibile interagire con gli altri studenti. In questo contesto dinamico, ciascun utente ha l'opportunità di condividere i propri oggetti creati, acquisirne di nuovi e interagire con gli altri partecipanti attraverso una chat integrata.

A differenza dei precedenti livelli, questa fase non è suddivisa in step distinti tanto meno introduce nuovi concetti teorici; piuttosto, si concentra sull'aspetto collaborativo e sulla condivisione con gli altri studenti, rispondendo direttamente al bisogno emerso nella fase di analisi degli articoli e riflessi nelle scelte progettuali adottate 3.1. Questa livello mira a favorire la collaborazione, l'interscambio di idee e la costruzione di un ambiente virtuale che promuova attivamente l'apprendimento sociale.

Nome	**Richiesta**	**Risposta**	**Cambiamento**
Creazione di un nuovo oggetto	In Action Menu, viene chiesto all'utente di muovere creare un nuovo oggetto	L'utente seleziona la funzionalità di creazione nel Side Menu; Definisce da quali forme geometriche o oggetti esistenti partire, trasformazioni ed attributi applicare, utilizzando delle funzionalità accessibili attraverso l'interfaccia grafica; Una volta completato, l'oggetto viene salvato e risulta accessibile nel Side Menu	Se l'utente aggiunge il nuovo oggetto nel Canvas, viene generato il codice contenente classe ed istanza della classe;

Tabella 3.7: Step del livello 6

Infine, nel **livello 6** l'utente può dare vita a nuovi oggetti partendo da una vasta gamma di forme geometriche e applicando varie trasformazioni, sfruttando

sia il codice che le funzionalità integrate nel gioco. Una volta completato il processo creativo, l'oggetto creato viene integrato nell'elenco degli oggetti disponibili nel Side Menu, permettendo all'utente di condividerlo con gli altri partecipanti, particolarmente nel livello 5. Questa livello è progettato per stimolare la creatività degli utenti e promuovere un ambiente che favorisce l'interscambio di idee e la collaborazione, sottolineando così l'importanza di condividere e costruire insieme nell'ambito del gioco Canvas Coding. Gli step associati a questo livello sono riportati in tabella 3.7.

Questo livello copre i seguenti argomenti:

- Ereditarietà e polimorfismo (4 Anno, UDA 1);
- Gerarchie di oggetti (4 Anno, UDA 1).

Dopo aver esaminato dettagliatamente tutti i livelli emersi nella fase di progettazione, è stata presa la decisione di implementare i primi tre livelli, in quanto offrono una copertura esaustiva delle unità di apprendimento insegnate nel terzo anno. Con l'aggiunta del quarto livello, si è notato che le UDA (Unità Didattiche d'Apprendimento) coperte rimangono sostanzialmente le stesse. Il quinto livello, al contrario, non presenta una mappatura diretta con nessuna UDA, mentre il sesto livello è focalizzato su una porzione limitata delle UDA del quarto anno. Tutti questi dettagli sono chiaramente evidenziati nella tabella sottostante.

UDA	Livello 1	Livello 2	Livello 3	Livello 4	Livello 6
3 Anno, UDA 1	SI	SI	SI	SI	NO
3 Anno, UDA 2	SI	SI	SI	SI	NO
3 Anno, UDA 3	SI	NO	NO	SI	NO
3 Anno, UDA 4	NO	NO	SI	NO	NO
4 Anno, UDA 1	NO	NO	NO	NO	SI
4 Anno, UDA 2	NO	NO	NO	NO	SI

Tabella 3.8: Copertura delle UDA

3.3 Prototipi

Un prototipo è una rappresentazione o implementazione concreta ma parziale di un progetto di sistema che consente di mostrare e percepire l'interattività e le funzionalità di base.

Il **prototipo a bassa fedeltà** espone le principali informazioni, interazioni e scelte progettuali con molti dettagli mancanti ed è realizzato a mano, consentendo

allo sviluppatore di concentrarsi sui concetti e sulle funzionalità principali, invece che sui piccoli dettagli.

Il **prototipo a media fedeltà** è utilizzato durante il processo di progettazione e sviluppo, come ad esempio un sito web, un'applicazione o un dispositivo, e consente di testare e validare concetti, flussi di lavoro e interazioni senza dedicare eccessive risorse alla creazione di dettagli fini.

3.3.1 Bassa fedeltà

Il prototipo è stato disegnato a mano utilizzando fogli di carta per rappresentare le schermate del gioco. Tutte le schermate sono state disegnate manualmente e rappresentano i vari stati raggiungibili, in particolare per i livelli dal primo al quarto. Il flusso di azioni eseguibili all'interno di ogni prototipo è indicato dal numero di fianco al cursore: seguendo il valore in ordine crescente, viene ricreato il flusso completo.

A seguire, per ogni livello implementato, è presentato il prototipo realizzato ed il flusso di operazioni da svolgere.

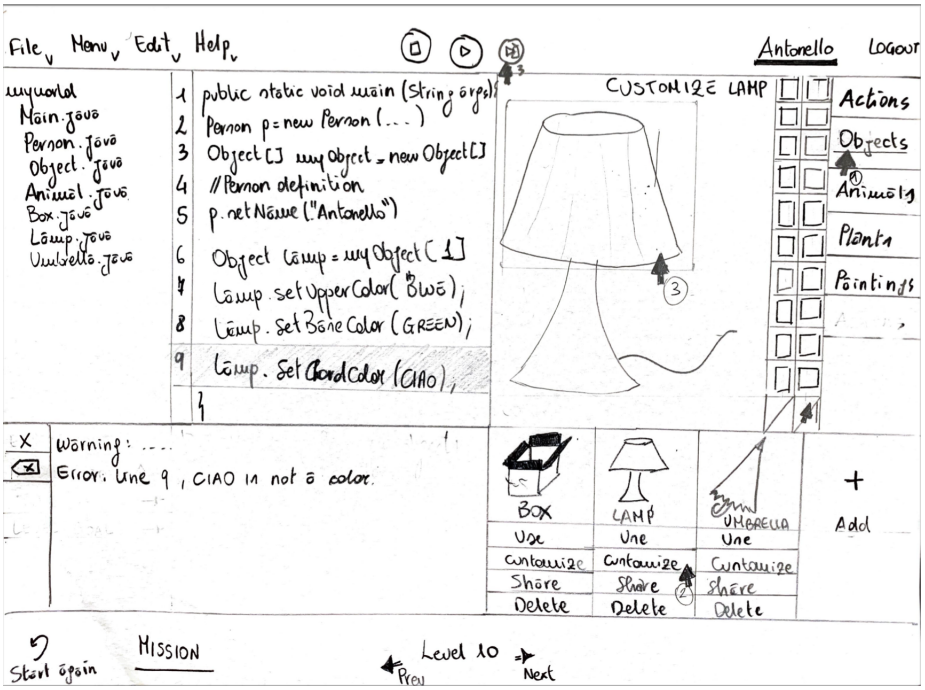

Figura 3.3: Prototipo a bassa fedeltà: Livello 1

Nell'immagine 3.3 è mostrato il flusso di operazioni da svolgere nel primo livello per completare tutti gli step previsti. In particolare:

1. **Selezione dell'oggetto da aggiungere**: l'utente seleziona "Object" dal Side Menu;

2. **Aggiunta dell'oggetto nel Canvas**;

3. **Personalizzazione dell'oggetto**: l'utente seleziona l'oggetto da personalizzare e ne modifica alcuni attributi, commettendo un errore;

4. **Compilazione del codice**: premendo sul pulsante di compilazione, viene mostrato l'output nella console e la riga di codice errata viene evidenziata.

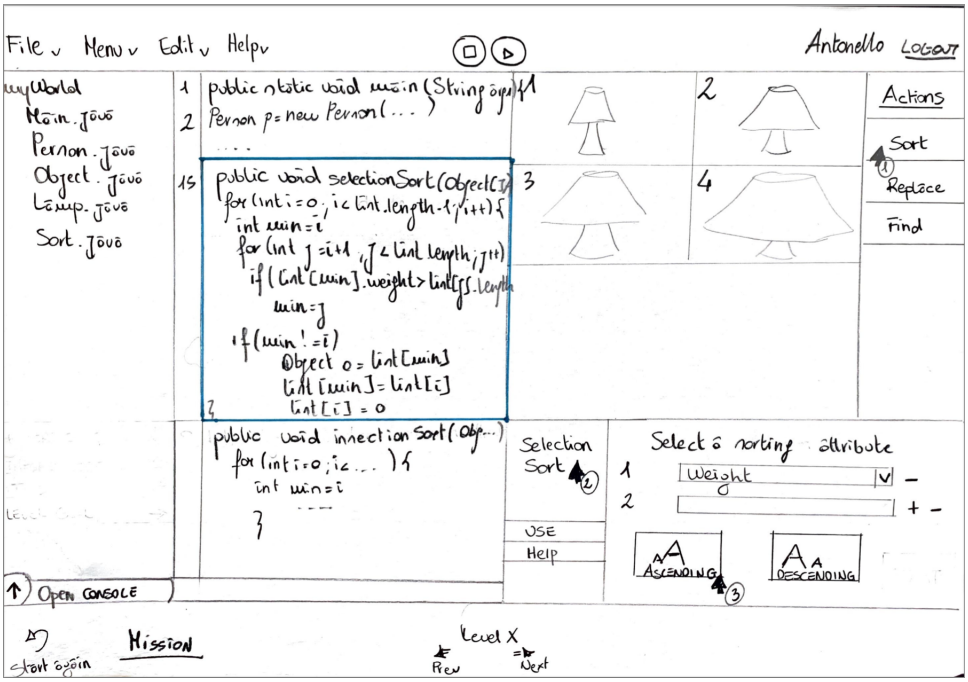

Figura 3.4: Prototipo a bassa fedeltà: Livello 2

Successivamente, in figura 3.4 è mostrato il flusso di operazioni da svolgere nel secondo livello, al fine di ordinare gli oggetti presenti nel Canvas e di generare il codice di ordinamento associato. Anche in questo caso, seguendo il numero di fianco al cursore, è possibile ricreare la sequenza di azioni:

1. **Ordinamento degli oggetti:** l'utente seleziona la voce "Sort" dal Side Menu. Questa, apre una schermata di dialogo che consente di definire le caratteristiche dell'ordinamento da svolgere;

2. **Definizione dell'algoritmo di ordinamento**: nella schermata viene mostrato la scelta del "Selection Sort[5]";

3. **Settaggio del tipo di ordinamento**: l'utente seleziona l'ordinamento crescente;

4. **Generazione del codice associato**: come evidenziato nel riquadro azzurro, viene generato il codice associato all'algoritmo di ordinamento scelto;

5. **Ordinamento grafico degli oggetti**: all'interno del Canvas, gli oggetti vengono disposti in ordine crescente a seconda della loro dimensione.

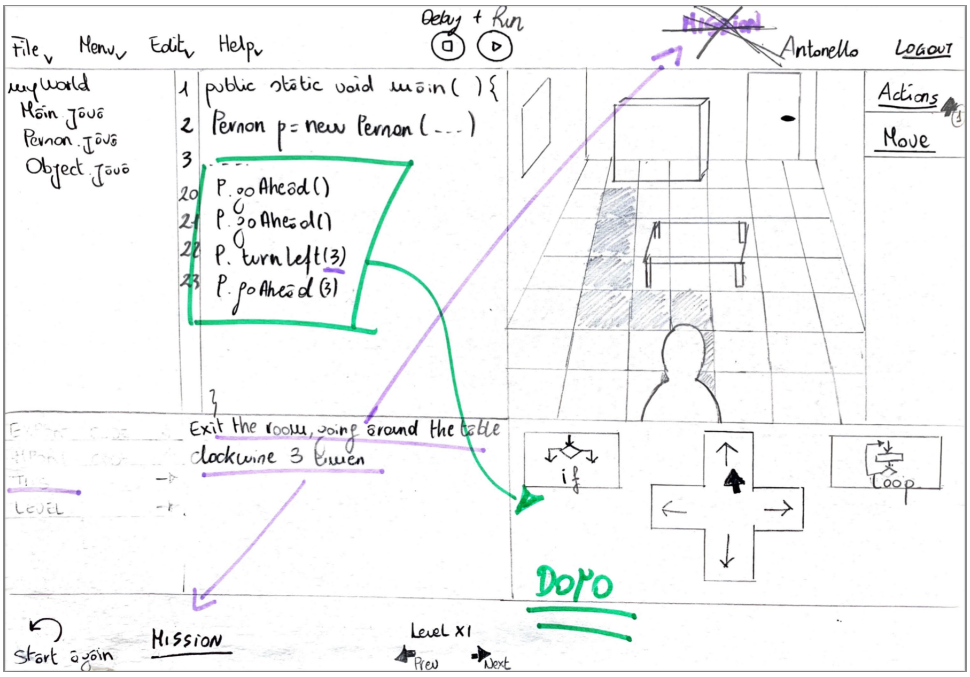

Figura 3.5: Prototipo a bassa fedeltà: Livello 3

A seguire, in figura 3.5 è indicato il flusso di operazioni che consentono all'avatar di spostarsi nel Canvas all'interno del terzo livello. La successione di azioni da eseguire è la seguente:

1. **Selezione funzionalità di movimento**: all'interno del Side Menu, l'utente seleziona la funzionalità "Move" che rende visibile il componente grafico di movimento;

[5]Noto algoritmo di ordinamento.

2. **Definizione del percorso**: mediante la pressione delle frecce direzionali nel componente di movimento, l'utente definisce il percorso che l'avatar dovrà percorrere. Ad ogni pressione, viene generato il codice associato.

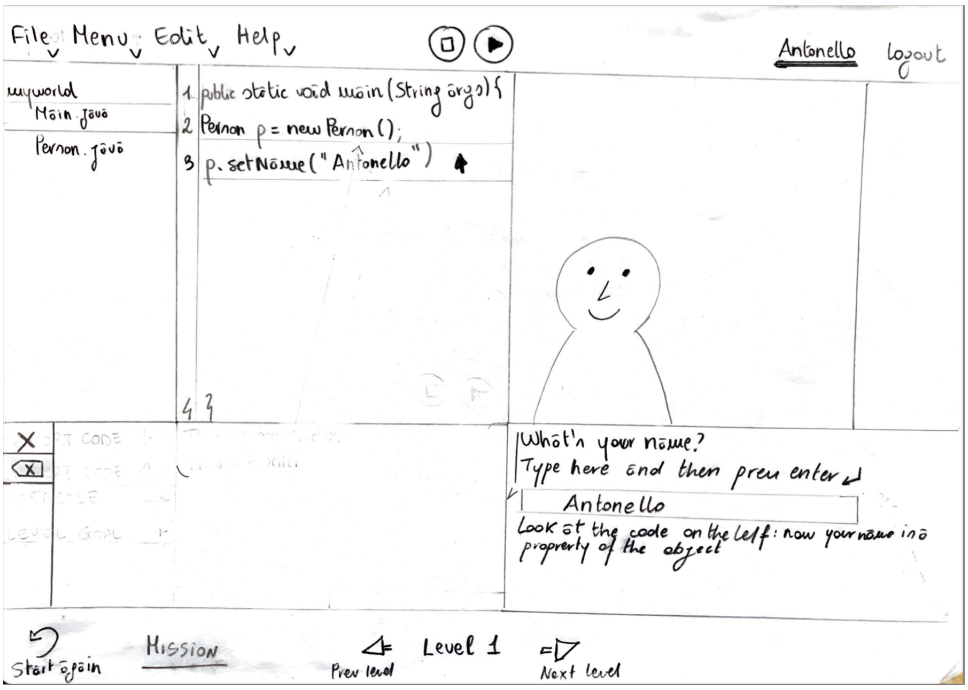

Figura 3.6: Prototipo a bassa fedeltà: Livello 4

Infine, nell'immagine 3.6 è mostrato il prototipo del livello 4: viene indicata la sequenza di operazioni da svolgere per completare il primo step previsto dal livello, ovvero la definizione di caratteristiche basilari per completare il profilo personale. A seguire, le operazioni svolte:

1. **Definizione del nome all'interno della schermata di dialogo**: l'utente inserisce il nome all'interno dell'interfaccia;

2. **Inserimento del nome nel codice**: successivamente all'inserimento, l'attributo viene aggiunto al codice sotto forma di parametro del metodo setter.

3.3.2 Media fedeltà

Per realizzare il prototipo a media fedeltà è stato utilizzato Figma[6], una piattaforma di progettazione e prototipazione che consente a team di progettisti, sviluppatori e

[6] `www.figma.com`

altre parti interessate di collaborare in tempo reale su sui progetti.

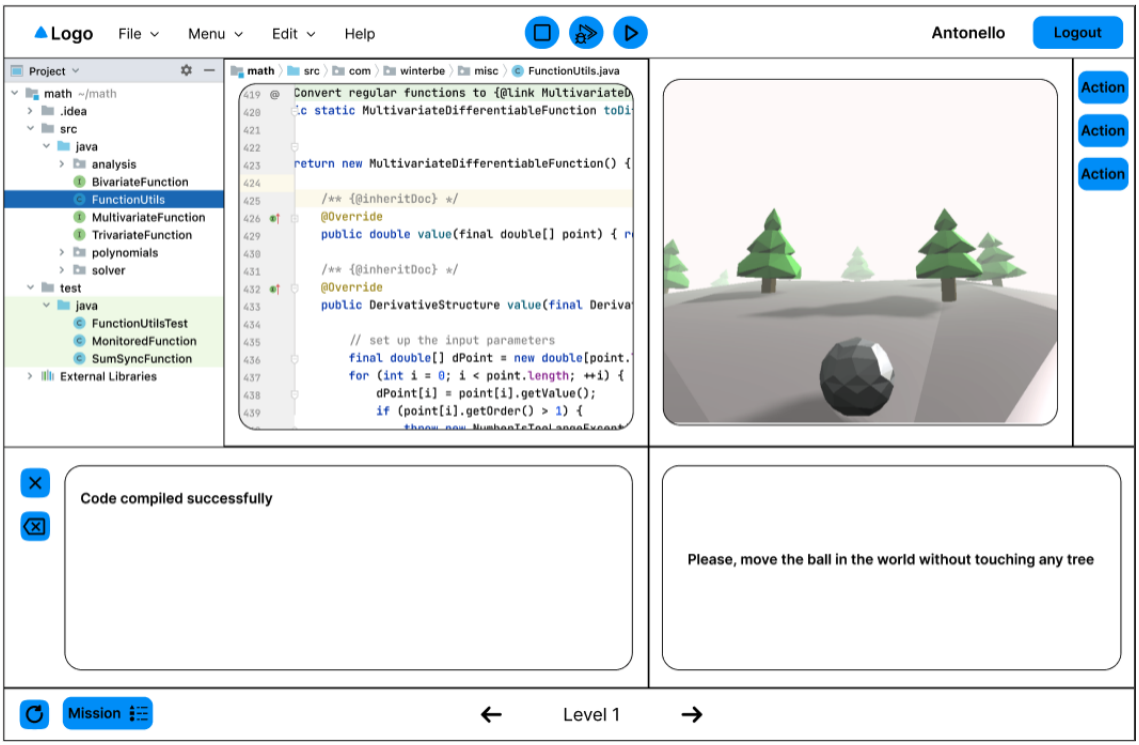

Figura 3.7: Prototipo a media fedeltà

Sfruttando questo strumento, è stato possibile creare il prototipo illustrato in figura 3.7, più ricco di dettagli e più definito rispetto ai prototipi precedenti. Pur avendo la possibilità, a livello di piattaforma, di sviluppare una sequenza di azioni finalizzate al completamento del livello in questione, per vincoli di tempo tale fase non è stata implementata. Tuttavia, va notato che questo prototipo, per la sua maggiore somiglianza grafica all'effettiva implementazione del gioco, fornisce un'anteprima più accurata e dettagliata del possibile risultato finale dell'intero processo di sviluppo.

3.4 Funzionalità di Canvas Coding

Per concludere, sono presentate le principali funzionalità previste dal gioco: per ognuna è indicato il componente logico al quale fa riferimento e le interazioni con l'utente. Inoltre, sono rappresentati degli schemi che riassumono graficamente le interazioni tra i componenti.

- **Interpretazione del codice**: Utilizzando il contenuto dell'editor di testo come punto di partenza, il processo di rigenerazione completa del Canvas è abilitato da un componente dedicato. Questo componente può essere associato all'atto di compilazione del codice, il quale può anche includere un controllo di correttezza della sintassi. Tuttavia, questa associazione non è obbligatoria, permettendo quindi di escludere il controllo sintattico in situazioni specifiche. In tal caso, l'esecuzione della compilazione influisce su tre componenti: il Codice, il Canvas e la Console. In scenari in cui il codice contiene errori, il componente grafico non viene generato e, al suo posto, viene visualizzato un avviso. Questa funzionalità, come si evince in figura 3.8, interconnette in modo dinamico i componenti Codice, Canvas e Console, assicurando una comunicazione integrata nel sistema e fornendo all'utente informazioni tempestive sulla correttezza del codice e sulla generazione dei componenti grafici.

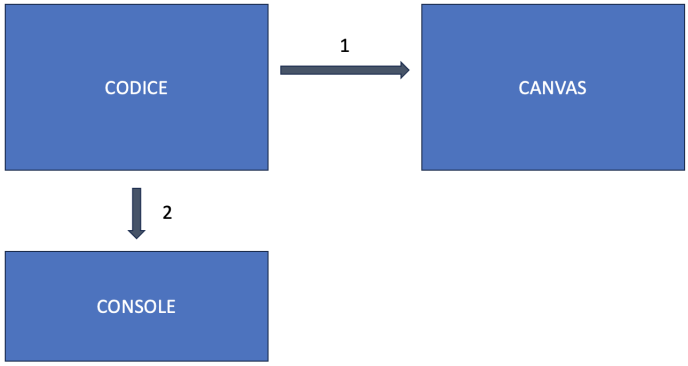

Figura 3.8: Componenti coinvolti nell'interpretazione del codice

- **Interpretazione Canvas**: processo inverso in cui il codice corrispondente viene generato a partire dal contenuto del Canvas. In questo scenario, il processo di interpretazione esclude la fase di compilazione del codice, poiché le azioni svolte sul Canvas generano sempre del codice sintatticamente corretto e privo di errori.

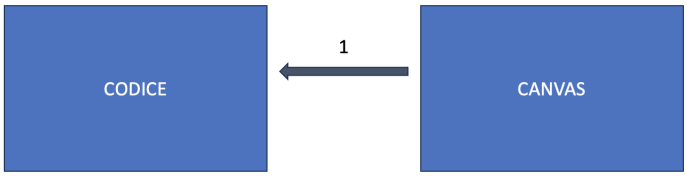

Figura 3.9: Componenti coinvolti nell'interpretazione del Canvas

Come si vede in figura 3.9 i componenti chiave coinvolti sono il Canvas e il Codice, con l'elemento distintivo che il punto di partenza è ora il Canvas stesso. Per ogni azione eseguita, viene automaticamente generato il codice corrispondente. All'interno del codice risultante, le righe interessate dalle modifiche apportate al Canvas sono evidenziate, fornendo così un chiaro collegamento tra le azioni grafiche e il codice sottostante. Questo approccio mira a rendere il processo di generazione del codice a partire dalle azioni sul Canvas intuitivo e visivamente trasparente.

Capitolo 4

Implementazione

Nel presente capitolo, è esaminata la selezione delle tecnologie adottate per l'implementazione del gioco Canvas Coding. Per ogni tecnologia, viene fornita una breve panoramica dello stato dell'arte, sono esplicitati i motivi che ne hanno guidato la scelta e sono forniti degli esempi estratti direttamente dal codice di Canvas Coding. Successivamente, ci si addentra nell'implementazione del gioco, suddividendo l'analisi tra front-end e back-end al fine di una comprensione più approfondita dei due approcci di sviluppo. Inoltre, per entrambe queste categorie, viene presentato il codice relativo all'implementazione di specifici componenti e funzionalità, seguito da una spiegazione dei dettagli tecnici e, ove possibile, arricchito da immagini esplicative che illustrano il processo. Questo approfondimento mira a offrire una visione completa e approfondita delle scelte tecnologiche e delle fasi di sviluppo del gioco.

4.1 Tecnologie utilizzate

A seguire, sono presentate le tecnologie utilizzate per l'implementazione del gioco. In particolare:

- **Front-end**: React e il framework MUI sono stati adoperati per l'implementazione della piattaforma online, Three.js, React Three Fiber e il plugin Drei sono stati utilizzati per implementare il Canvas e la visualizzazione 3D degli oggetti. Infine, React Ace è stato utilizzato per l'implementazione dell'editor di testo;

- **Back-end**: per la realizzazione del Server è stato utilizzato Spring Boot in Kotlin, per la memorizzazione dei dati Postgres e per l'autenticazione degli utenti Keycloak. Inoltre, al fine di favorire la condivisione della piattaforma, il tutto è stato inserito all'interno di un docker compose, assieme al front-end.

4.1.1 React e MUI

React[1] è una libreria JavaScript open-source utilizzata per la creazione di interfacce utente per applicazioni web. Sviluppata e mantenuta da Facebook[2], React è ampiamente utilizzata per la creazione di componenti riutilizzabili e per la gestione dello stato dell'interfaccia utente. A seguire, le principali caratteristiche di questa libreria:

- **Componenti**: le interfacce utente sono suddivise in componenti, intesi come dei blocchi modulari autonomi. Questi possono contenere della logica e rappresentare delle parti specifiche dell'interfaccia utente. Mediante l'utilizzo di questo approccio, è favorito il riuso del codice e la manutenibilità, in quanto basta fornire una sola implementazione del componente e richiamarla all'interno del codice quando richiesta. A seguire, un esempio di componente estratto dal codice di Canvas Coding.

```
1  export function HighlightMesh(props) {
2      const { position, overlap, overlapColor } = props;
3      return (
4          <mesh position={[position.x, 0, position.z]} rotation
       ={[-0.5 * Math.PI, 0, 0]} >
5              <planeGeometry args={[1, 1]}/>
6              <meshBasicMaterial color={overlap ? overlapColor :
       0x0000FF} side={THREE.DoubleSide}  depthTest={true} />
7          </mesh>
8      )
9  }
10
```

Listing 4.1: Esempio di componente React.js estratto dal codice del gioco

- **JSX (JavaScript XML)**: estensione di JavaScript che consente di scrivere codice simile all'HTML[3] all'interno, semplificando la definizione della struttura dell'interfaccia utente. In 4.1 è fornito un esempio

- **Virtual DOM**: quando lo stato di un componente cambia, React crea una rappresentazione virtuale del DOM [4] e la confronta con la rappresentazione effettiva. Questo permette a React di aggiornare solo le parti del DOM che sono state effettivamente modificate, garantendo delle performance maggiori.

[1] `it.legacy.reactjs.org`

[2] `www.facebook.com`

[3] Hyper Text Markup Language

[4] Document Object Model

- **Data Flow Unidirezionale**: i dati seguono un flusso unidirezionale, gli stati vengono passati come proprietà ai componenti figli, e qualsiasi modifica allo stato viene gestita da funzioni specifiche, garantendo una gestione chiara e prevedibile dello stato dell'applicazione. In 4.1, mediante il valore props, viene gestito il passaggio delle proprietà.
- **Hooks**: sono delle funzioni che consentono di utilizzare lo stato ed altre funzionalità come se fossero dei componenti basati su funzioni, anziché dei componenti di classe. In questo modo, rendono il codice più leggibile e consentono di utilizzare lo stato e i cicli di vita dei componenti in un contesto di funzione. A seguire, un esempio estratto dal front-end di Canvas Coding.

```
1     React.useEffect(() => {
2         if (!compiling && !loading) {
3             setCode(CanvasInterpreter(objects, sortParams,
    step?.id >= 7 ? true : false));
4         }
5     }, [objects])
6
```

Listing 4.2: Esempio di React Hook estratta dal codice del gioco

Per la realizzazione del front-end si è scelto di utilizzare questa tecnologia in quanto è una scelta popolare per lo sviluppo di applicazioni web moderne e scalabili. La sua sintassi dichiarativa, la gestione efficiente dello stato e la facilità di creazione di componenti riutilizzabili lo rendono un ottimo candidato nella scelta di librerie per lo sviluppo web.

```
1  <Box sx={{ display: 'flex', flexDirection: 'row', minHeight: '100
       vh' }}>
2      <CustomDialog ...}></CustomDialog>
3          <Box sx={{ flex: 1, display: 'flex' }}>
4              <Grid container >
5                  {/* Code and console */}
6                  <Grid item xs={7} >
7                      <CodeContainer ... />
8                  </Grid>
9                  {/* Canvas, side and action menu */}
10                 <Grid item xs={5} sx={{ display: 'flex',
       flexDirection: 'column', position: 'relative', height: '100%'
       }}>
11                     <CanvasContainer ... />
12                 </Grid>
13             </Grid>
14         </Box>
15 </Box >
```

Listing 4.3: Esempio di componente MUI estratto dal codice del gioco

Material-UI (MUI) è un framework di React basato sul design system Material Design di Google. Questo, lanciato nel 2014, si basa su principi di design chiari, elementi visivi e interattivi, e un approccio olografico alla progettazione dell'interfaccia utente. MUI offre un insieme di componenti React predefiniti e stilizzati secondo le linee guida del Material Design, come pulsanti, caselle di testo, barre di navigazione, carte, modali e molti altri. In 4.3 è possibile vedere un utilizzo del Grid System, ovvero un sistema di griglie flessibile basato su Flexbox, che facilita la creazione di layout responsivi e adattabili.

4.1.2 Three.js, React Three Fiber e Drei

Three.js è una libreria JavaScript open-source utilizzata per la creazione di grafica 3D interattiva, che semplifica la programmazione 3D utilizzando WebGL e consentendo agli sviluppatori di creare esperienze immersive direttamente all'interno di pagine web. Three.js facilita la creazione di scene 3D complete mediante l'inserimento di oggetti, luci, ombre e materiali. Caratteristiche della libreria sono:

- **Telecamere e controlli**: Three.js offre il supporto a diverse tipologie di telecamere (prospettiva, ortografica) e controlli, che consentono di semplificare la navigazione all'interno delle scene 3D. Queste consentono di interagire con gli oggetti, zoomare e ruotare la visuale;
- **Geometrie predefinite**: la libreria include diverse geometrie predefinite (cubi, sfere, piani) e materiali (colori, texture, luci) che possono essere utilizzati per creare oggetti 3D complessi e dettagliati, come nell'esempio seguente.

```
  const geometry = createGeometry(type, size);
  const material = new THREE.MeshBasicMaterial(color ? {
    color
  } : {
    map: THREE.TextureLoader(texture)
  });
  const mesh = new THREE.Mesh(geometry, material);
  const obj = new THREE.Object3D();

  obj.add(mesh);
)
```

Listing 4.4: Esempio di geometrie predefinite in Three.js

- **Prestazioni elevate**: grazie a WebGL, la libreria sfrutta tutta la potenza della grafica hardware del dispositivo;
- **Supporto dei formati OBJ, STL, e glTF**: gli sviluppatori possono integrare facilmente i modelli 3D creati con strumenti di modellazione tridimensionale

all'interno delle loro scene. Questa caratteristica è stata utilizzata per realizzare un modello 3D di un coniglio utilizzando Blender[5], utilizzato all'interno del livello 3.

```
1  assetLoader.load(rabbitUrl.href, function (gltf) {
2        const model = gltf.scene;
3        model.position.set(3, 1, 0)
4        scene.add(model);
5        mixerRabbit = new THREE.AnimationMixer(model);
6
7        const clips = gltf.animations;
8
9        const clip = THREE.AnimationClip.findByName(clips, '
     SnoutTailEarsAction');
10       const action = mixerRabbit.clipAction(clip);
11       action.play();
12
13    }, undefined, function (error) {
14       console.error(error);
15    })
```

Listing 4.5: Esempio di utilizzo di un modello 3D in formato gLTF in Three.js

Grazie alla sua versatilità e alla facilità di utilizzo, Three.js è uno degli strumenti più popolari per lo sviluppo di contenuti 3D sul web: è molto utilizzato per la creazione di giochi, visualizzazioni scientifiche, esperienze interattive e altre applicazioni.

React Three Fiber (R3F) è una libreria React che offre un'interfaccia dichiarativa per la creazione di grafica 3D utilizzando Three.js. In sostanza, R3F integra la potenza di Three.js, con il modello di programmazione dichiarativo di React. A seguire è fornita l'implementazione dell'importazione del modello 3D del coniglio, come in 4.5, utilizzando l'approccio modulare di React.

```
1  export function RabbitModel(props) {
2      const rabbitUrl = new URL('../Assets/Rabbit.glb', import.meta.
     url);
3      const gltf = useLoader(GLTFLoader, rabbitUrl.href);
4      const [modelRef] = useYuka({ type: Vehicle, ...props });
5      const mixerRef = React.useRef();
6
7      React.useEffect(() => {
8          if (!mixerRef.current) {
9              mixerRef.current = new THREE.AnimationMixer(gltf.scene
     );
```

[5]Programma di modellazione 3D.

```
10          }
11          ....
12      }, []);
13
14      useFrame((state, delta) => {
15          if (mixerRef) {
16              mixerRef?.current?.update(delta);
17          }
18      });
19
20      return <primitive object={gltf.scene} ref={modelRef} />;
21  }
```

Listing 4.6: Esempio di utilizzo di un modello 3D in formato gLTF in React Three Fiber

R3F consente di dichiarare la scena 3D, gli oggetti e le luci utilizzando una sintassi simile a JSX, rendendo il codice più leggibile e manutenibile. I componenti 3D possono essere facilmente composti per creare scene complesse. Inoltre, fornisce un hook personalizzato chiamato useThree che offre accesso al contesto Three.js. Come si vede in 4.7, questo hook semplifica l'interazione con Three.js e permette di accedere facilmente a funzionalità come la camera, la scena e il renderer.

```
1  export function Scene(props) {
2      const { selectObj, .....c} = props;
3      const { camera, scene, gl } = useThree();
```

Listing 4.7: Esempio di utilizzo dell'Hook useThree

Assieme a R3F è stato utilizzato **Drei**, una libreria che fornisce un insieme di componenti React ad alto livello che semplificano ulteriormente lo sviluppo di grafica 3D, utilizzando Three.js all'interno di applicazioni React. Caratteristiche di R3F sono:

- **Supporto alle animazioni**: Drei semplifica l'inserimento di animazioni agli oggetti 3D, consentendo agli sviluppatori di integrare movimenti e transizioni nella grafica 3D;

- **Integrazione con R3F**: Drei è progettato per lavorare in simbiosi con React Three Fiber, in quanto entrambe le librerie sono basate su React e Three.js.;

- **Componenti ready-to-use**: Drei offre una serie di componenti pronti all'uso che consentono di creare elementi comuni in una scena 3D, come luci, camere, ambienti, controlli per la navigazione.

4.1.3 React Ace

React-ace è un componente React che agisce da wrapper ad un editor di testo open-source chiamato Ace scritto in JavaScript che offre funzionalità avanzate per la modifica di codice, come evidenziazione della sintassi (linter), completamento automatico, generazione a partire da esempi (snippets) e molto altro. Questo componente semplifica l'integrazione dell'editor Ace in applicazioni React, esattamente come R3F e Three.js.

Durante la fase di ricerca sono stati identificati altri editor, come React-codemirror e Monaco editor. Per ciascuno di questi, sono esposti in tabella 4.1 i vantaggi e gli svantaggi individuati, che hanno infine orientato la scelta a favore di React-ace.

Nome editor	Pro	Contro
React-ace	Snippets	No linter
React-codemirror	Linter	No Snippets
Monaco editor	Linter	No Snippets e No supporto Java

Tabella 4.1: Confronto tra editor di codice per React

Sebbene React-ace non abbia un supporto ufficiale per il linting, questa funzionalità è stata implementata manualmente come si vede in 4.8. Grazie a questa funzione, viene evidenziata la riga contenente l'errore e indicazioni specifiche sull'errore commesso sono fornite mediante l'utilizzo di annotazioni.

```
1  React.useEffect(() => {
2      if (!loading) {
3      const session = editorRef.current.getSession();
4      removeMarkers();
5      changedLines.forEach((change) => {
6      const newRange = new Range(change.start, 0, change.end, 0);
7      session.addMarker(newRange, 'ace_selection', 'text', false);})
       ;
8      }
9  }, [changedLines]);
10 const removeMarkers = () => {
11     if (editorRef.current) {
12         const session = editorRef.current.getSession();
13         const markers = session.getMarkers();
14         for (const key in markers) {
15             if (markers[key].clazz === 'ace_selection') {
16                 session.removeMarker(key);
17             }}}}
```

Listing 4.8: Implementazione del linting per React-ace

4.1.4 Spring boot

Spring Boot è un framework open-source per lo sviluppo di applicazioni Java e Kotlin basato sul Framework Spring, che semplifica lo sviluppo di applicazioni web, offrendo una configurazione automatica, una gestione delle dipendenze semplificata garantendo uno sviluppo rapido. Caratteristiche di questo framework sono:

- **"Conventions over configurations"**: utilizzando questa filosofia che favorisce la convenzione rispetto alla configurazione, molti settaggi di base sono gestiti automaticamente, riducendo la necessità di configurazioni personalizzate;
- **Servers incorportati**: Tomcat, Jetty o Undertow, sono esempi di server già inclusi in Spring Boot. In questo modo, è possibile eseguire facilmente un'applicazione Java senza dover ricorrere ad un server esterno.
- **Ecosistema**: Spring Boot è progettato per funzionare bene con le altre tecnologie dell'ecosistema Spring, come MVC per lo sviluppo di applicazioni web, Spring Data per l'accesso ai dati, e Spring Security per la sicurezza. A seguire, un esempio di utilizzo di Spring Security all'interno del gioco.

```
1  @EnableWebSecurity
2  class WebSecurityConfig(private val jwtAuthConverter:
       JwtAuthConverter) {
3
4      @Bean
5      @Throws(Exception::class)
6      fun securityFilterChain(http: HttpSecurity):
       SecurityFilterChain {
7          http.csrf().disable().authorizeHttpRequests().
       requestMatchers(HttpMethod.POST, "/API/login").permitAll()
8          http.authorizeHttpRequests()
9              // Profile endpoints
10             .requestMatchers(HttpMethod.GET, "/API/profiles").
       hasRole(TEACHER)
11             .requestMatchers(HttpMethod.GET, "/API/profile").
       authenticated()
12             .requestMatchers(HttpMethod.DELETE, "/API/profiles
       /{email}").hasRole(TEACHER)
```

Listing 4.9: Utilizzo di Spring Security nel codice del gioco

4.1.5 Keycloak e PostgreSQL

Keycloak è una piattaforma open-source per la gestione degli accessi e delle credenziali utente, sviluppata da Red Hat. Si focalizza sulla sicurezza dell'identità,

consentendo alle applicazioni di delegare l'autenticazione e l'autorizzazione degli utenti a un server centralizzato.

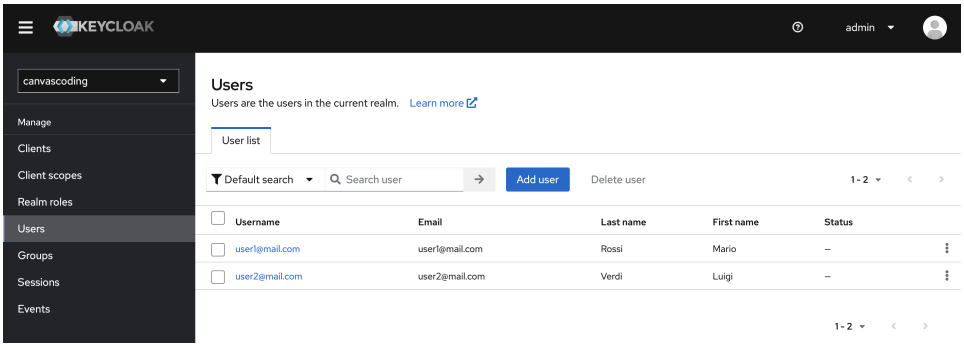

Figura 4.1: Management console di Keycloak per Canvas Coding

A seguire, alcune caratteristiche di Keycloak che hanno condizionato l'utilizzo nella realizzazione del gioco:

- **Gestione delle credenziali**: Keycloak gestisce la memorizzazione sicura delle credenziali dell'utente, compresi la registrazione, il reset della password e la gestione delle tipologie di utenze;
- **Gestione delle Sessioni**: Monitora e gestisce le sessioni utente, consentendo agli amministratori di visualizzare e revocare le sessioni attive;
- **Standard di Sicurezza**: il supporto degli standard di sicurezza come OAuth 2.0 e OpenID Connect, utilizzati per l'autenticazione e l'autorizzazione, garantiscono la massima sicurezza dell'accesso alle risorse.

PostgreSQL, spesso noto come Postgres, è un sistema di gestione di database relazionale (RDBMS) open-source scalabile ed estensibile. Utilizzato in applicazioni di vario genere, dai progetti open-source alle grandi imprese, grazie alla sua affidabilità, robustezza e flessibilità. La sua versatilità lo rende adatto per una vasta gamma di scenari di utilizzo, tra cui applicazioni web, sistemi di gestione dei dati geografici, e molto altro. A seguire, le principali caratteristiche e funzionalità:

- **SQL Standard**: PostgreSQL implementa gli standard SQL ANSI/ISO, offrendo una conformità elevata e garantendo la portabilità delle applicazioni tra database relazionali;
- **Integrità dei Dati e Vincoli**: supporta i vincoli di integrità come chiavi primarie, chiavi esterne, vincoli di unicità per garantire la coerenza e l'integrità dei dati nel database;

- **Transazioni ACID**: PostgreSQL garantisce il supporto per le transazioni ACID (Atomicità, Consistenza, Isolamento, Durabilità), fondamentali per garantire la coerenza e l'affidabilità dei dati in ambienti transazionali.

4.1.6 Docker

Docker è una piattaforma open-source che semplifica la creazione, la distribuzione e l'esecuzione di applicazioni. Queste sono inserite all'interno di un container, un'unità leggera e portatile che contiene tutto il necessario per l'esecuzione di un'applicazione, inclusi il codice, le librerie e le dipendenze. Caratteristiche di Docker sono:

- **Containerizzazione**: utilizza la tecnologia di "containerizzazione", consente di isolare le applicazioni e i loro ambienti di esecuzione. I container possono essere eseguiti su qualsiasi piattaforma che abbia Docker installato, indipendentemente dal SO o processore;

- **Immagini Docker**: le applicazioni sono distribuite mediante delle immagini, ovvero dei pacchetti leggeri e autosufficienti che contengono tutto il necessario per eseguire un'applicazione. Ad ogni immagine è associato un numero di versione, ed inoltre queste possono essere distribuite attraverso registri come Docker Hub;

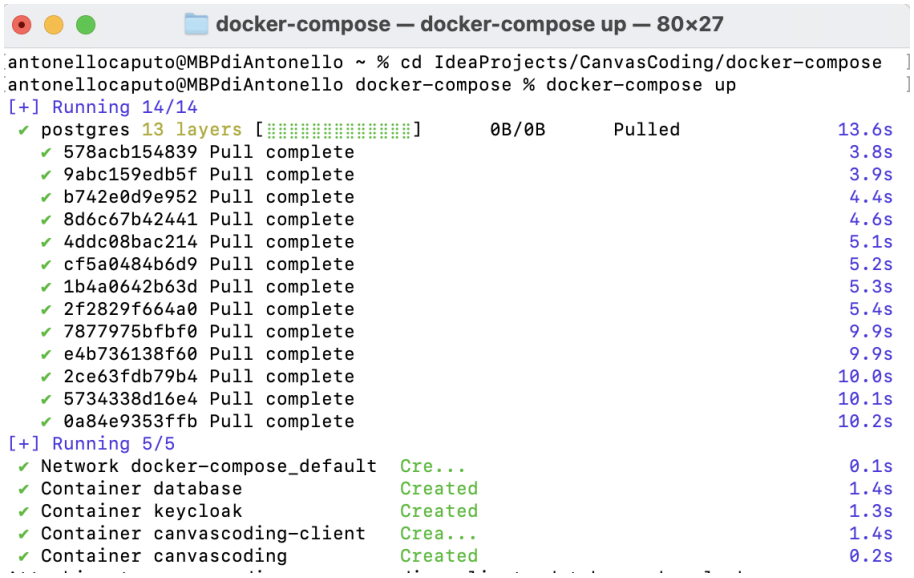

Figura 4.2: Avvio di Canvas Coding da terminale mediante Docker compose

- **Docker Compose**: strumento che semplifica la definizione e la gestione di servizi Docker composti da diversi container. Consente di definire un'applicazione multi-container in un file YAML e di avviare/fermare facilmente l'applicazione con un singolo comando. Ad esempio, Canvas Coding si basa su 4 container diversi: uno per il server Spring Boot, uno per il database Postgres, uno per Keycloak ed infine uno per il front-end React. Mediante la definizione di un docker-compose, è possibile eseguire/fermare l'esecuzione di tutti i container con un solo comando, come si vede in 4.2.

4.2 Sviluppo

Proseguendo, sono dettagliati i componenti chiave del gioco, e per ciascuno di essi, sono forniti degli estratti significativi dalle rispettive implementazioni. Inoltre, dove possibile, delle schermate rappresentanti il componente sviluppato sono proposte.

4.2.1 Front-end

In figura 4.3 è rappresentata l'implementazione del front-end del gioco, sviluppato utilizzando React e MUI.

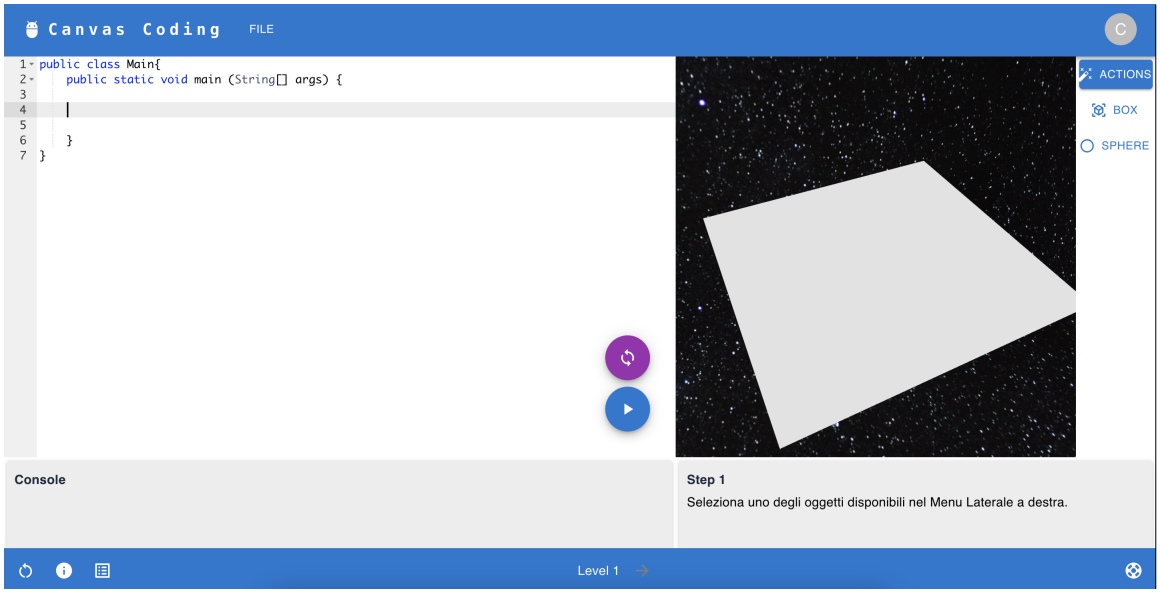

Figura 4.3: Front-end in React del gioco Canvas Coding

All'interno del componente *Homepage*, renderizzato al completamento del processo di autenticazione, sono richiamati tutti i principali elementi del gioco, tra cui

Canvas, Editor, Side Menu e Action Menu. Questo componente è stato sviluppato utilizzando la flessibile struttura a griglia (Grid Layout) di MUI, che garantisce una disposizione ottimizzata del un contenuto in grado di adattarsi efficacemente alle dimensioni dello schermo. Nel codice 4.10 in seguito, viene presentata l'implementazione di Homepage.js, accompagnata da un dettaglio approfondito di come questi componenti chiave siano integrati nel contesto della pagina principale del gioco.

```
return (
    <Box sx={{ display: 'flex', flexDirection: 'row', minHeight: '100vh' }}>
        {/* Dialog */}
        <CustomDialog ...props></CustomDialog>
    <Box sx={{ flex: 1, display: 'flex' }}>
        <Grid container >
        {/* Code and console */}
        <Grid item xs={7} >
            <CodeContainer ...props />
            </Grid>
        {/* Canvas, side and action menu */}
        <Grid item xs={5} sx={{ display: 'flex', flexDirection: 'column', position: 'relative', height: '100%' }}>
            <CanvasContainer ...props />
            </Grid>
        </Grid>
    </Box>
</Box >
    );
```

Listing 4.10: Implementazione di Homepage.js

Diversi Hooks sono inseriti all'interno di questo componente, ognuno in ascolto sui cambiamenti dei principali attributi del gioco. In particolare, quelli relativi alla compilazione e sincronizzazione del codice ed alla interpretazione del Canvas sono inseriti in questo componente.

La funzionalità di **compilazione del codice** svolge le seguenti operazioni:

1. **Interpreta il codice**, generando i componenti grafici da mostrare nel Canvas: questa operazione viene svolta mediante l'interprete sviluppato appositamente per il gioco;

2. **Compila il codice** mediante una chiamata ad un servizio esterno di compilazione, sfruttando le API fornite da servizio chiamato Judge-0[6]: inserendo nel corpo della chiamata il contenuto del codice Java, il servizio restituisce

[6] `www.judge0.com`

dopo qualche secondo l'esito della compilazione, l'output della console e gli eventuali errori generati. Questo consente di evitare l'implementazione di un server di compilazione locale, anche se questo servizio esterno presenta diverse limitazioni, come ad esempio un numero massimo di operazioni per giorno;

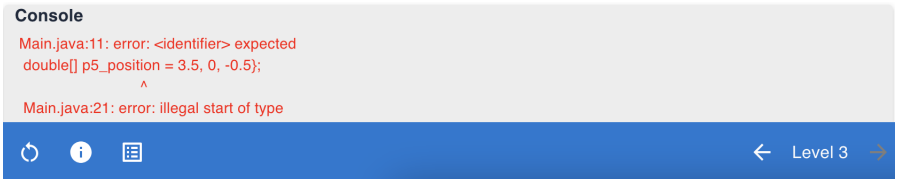

Figura 4.4: Esito della compilazione mostrato nella console di output

3. **Evidenzia il codice contenente errori**: come indicato precedentemente, l'editor React-ace non include la funzionalità di linting built-in. Per ovviare a questa limitazione è stata implementata una funzione che, in base al risultato della compilazione, sottolinea le righe contenenti errori ed inserisce delle annotazioni di fianco al codice, come mostrato in figura 4.5. Questa funzionalità non è disponibile a run-time ma solo a compile-time, ma consente all'utente di individuare efficacemente gli errori commessi.

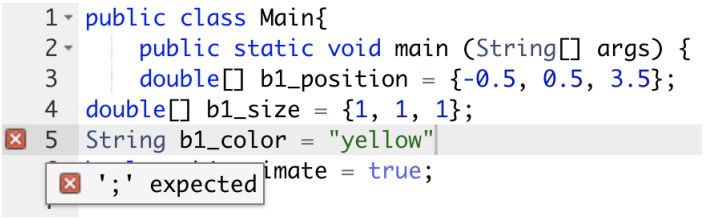

Figura 4.5: Linting a compile-time in React-ace

La funzionalità di *sincronizzazione*, invece, è accessibile mediante il pulsante Sync in viola nell'editor ed effettua solo l'interpretazione del codice. Questo interprete, a partire dal codice Java, genera un array di oggetti che viene successivamente fornito al componente Canvas per la visualizzazione.

```
1 export function CodeInterpreter(javaCode, pathInterpreter) {
2
3 const objectDeclarations = javaCode.match(/(\w+)\s+(\w+)\s+=\s+new
    \s+(\w+)\(([^)]+)\);/g);
4
5 const variableDeclarations = javaCode.match(/(\w+(?:\[\])?)\s+(\w
    +)\s*=\s*((?!new\s+)[^;]+);/g);
6
```

```
7  const variables = variablesFromDeclaration(variableDeclarations);
8  const objectsArray = objectsFromDeclaration(objectDeclarations,
       variables, pathInterpreter);
9
10 return objectsArray;
11 });
```

Listing 4.11: Funzione codeInterpreter richiamata da Hooks in Homepage.js

Come mostrato in 4.11, mediante l'utilizzo di numerose e complesse regEx[7], l'interprete è in grado di:

1. **Separare le dichiarazioni degli oggetti dal codice**: estrae le dichiarazioni di oggetti, espresse mediante word, dalle righe del codice che presentano la sintassi "(word) 1..Nspace (word) 1..Nspace = 1..Nspace "new" 1..Nspace (word)", dove 1..Nspace indica 1 o più spazi;

2. **Separare le dichiarazioni delle variabili dal codice**: estrae le variabili che seguono la sintassi "(word) 1..Nspace (word) 0..Nspace = 0..Nspace", dove 0..N space indica 0 o più spazi;

3. **Generare un array di variabili ed uno di oggetti**: partendo dagli oggetti estratti, viene generato un array di oggetti, ciascuno avente come attributi le variabili estratte.

L'array di variabili viene generato nella funzione 4.12, estraendo le singole variabili contenute all'interno del codice.

```
1  function variablesFromDeclaration(variableDeclarations){
2      let variables = [];
3      if(variableDeclarations){
4      variableDeclarations.forEach( declaration => {
5      const clear = declaration.match(/^(?!.*\bnew\b).*$/);
       // RegEx -> remove instances containing "new"
6          if(clear){
7          clear.forEach(variable =>{
8          const split = variable.match(/(\w+(?:\[\])?)\s+(\w+)\s*=\s
       *([^;]+);/); // RegEx-> split variable declaration in parts (
       type, name, value)
9              if(split){
10                 try{
11                     validateVariable(split);
12                     const newvar = new Variable(split[2], split
       [1], valueFromVariable(split[1],split[3]))
13                     variables.push(newvar);
```

[7]Un'espressione regolare è una sequenza di simboli che identifica un insieme di stringhe.

```
14                      }catch(err){
15                      alert("CodeInterpreter Invalid data: " + err.
    message);           // Invalid data: No property: name
16                      }}
17              });
18          }})}
19      return variables;}
```

Listing 4.12: Funzione variablesFromDeclaration richiamata da CodeInterpreter

La funzione qui presente svolge le seguenti operazioni:

1. Applica l'operatore *match* per rimuovere le istanze contenenti la parola "new";
2. Per ogni elemento nell'array generato, lo suddivide in oggetti composti dai seguenti attributi: tipo, nome, valore;
3. Applica una funzione di validazione delle variabili, che verifica che tutti i campi necessari siano inseriti;
4. Estrae il valore dalla variabile in base al tipo, utilizzando la funzione *valueFromVariable*;
5. Inserisce le variabili all'interno dell'array.

Infine, la funzione *objectsFromDeclaration* consente di generare un array di oggetti a partire dalle dichiarazioni estratte dal codice: a seguire è fornita l'implementazione.

```
1  function objectsFromDeclaration(objectDeclarations, variables,
       pathInterpreter){
2      let objectArray = []
3      if (objectDeclarations) {
4          objectDeclarations.forEach(declaration => {
5              const matches = declaration.match(/(\w+)\s+(\w+)\s+=\s
       +new\s+(\w+)\(([^)]+)\);/);
6              if (matches) {
7                  try{
8                      let newObj = {
9                          id: matches[2].replace(/\D/g, '')*1,
10                         type: matches[3].toUpperCase()
11                     }
12                     if(matches[4]){
13                     matches[4].split(',').forEach( param => {
       const key = param.trim().match(/_(.+)/)[1];
14                     const value = variables.find( value => value.
       name === param.trim()).value
15                     newObj[key] = value
16                     })
```

```
17                  }
18                  objectArray.push(newObj)
19                  }catch(err){
20                  alert("CodeInterpreter Invalid data: " + err.
    message); // Invalid data
21                  }
22              }
23          });
24  }
```

Listing 4.13: Funzione objectsFromDeclaration richiamata da CodeInterpreter

In particolare, la funzione 4.13 svolge le seguenti operazioni:

1. Estrae la dichiarazione degli oggetti che seguono la sintassi “(word) 1..Nspace (word) 1..space = 1..Nspace "new" 1..Nspace (word)”. Ad esempio, se objectDeclarations contiene “Box b1 = new Box(b1_position, b1_size, b1_color);”, allora vengono estratti i seguenti campi: ’Box’, ’b1’, ’Box’, ’b1_position’, ’b1_size’, ’b1_color’;

2. Definisice l’id ed il tipo dell’oggetto a partire da quanto estratto precedentemente;

3. Suddivide i campi contenuti nel codice come ’b1_position, b1_size, b1_color’ in un array di 3 elementi e per ognuno di questi, cerca il corrispondente valore nell’array contente le variabili estratte nella funzione 4.12.

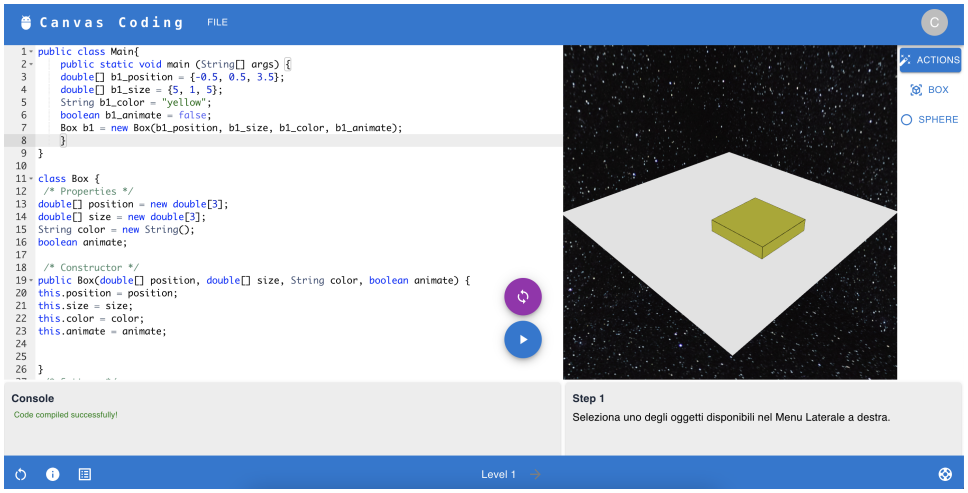

Figura 4.6: Interpretazione del codice e rappresentazione di un rettangolo in Canvas Coding

Un esempio concreto è rappresentato dalla definizione della classe “Box” e dall’inserimento di una sua istanza all’interno della classe main. Avviando il

processo di interpretazione attraverso l'utilizzo dei pulsanti di sincronizzazione o di compilazione, è possibile visualizzare l'oggetto creato direttamente nel Canvas, come illustrato nella figura 4.6. In questo modo gli utenti possono tradurre rapidamente le loro dichiarazioni di classi ed oggetti in elementi grafici tangibili all'interno del gioco.

Approccio inverso è quanto fornito dalla funzionalità di **interpretazione del Canvas** e degli elementi grafici in esso contenuti e generazione del codice associato. La funzione "CanvasInterpreter", partendo da un array di oggetto ricevuto come parametro d'ingresso, è in grado di generare il codice contenete: dichiarazioni delle variabili, chiamate a metodi, dichiarazione delle istanze e definizione delle classi.

```
1  export function CanvasInterpreter(objects, sortParams, reached) {
2      let instances = '';
3      let variables = '';
4      let methods = '';
5      let classes = '';
6      let calls = '';
7
8  variables = objects.map(element => {
9      try{
10         validateObject(element);
11         return objectToVariables(element)
12     }catch (err){
13         alert("CanvasInterpreter Invalid data: " + err.message);
               property: name
14         return '';
15     }
16 }).join('\n')
17
18     instances = objects.filter(obj => obj.type !== 'PATH').map(
       element => {
19         return objectToInstance(element)
20     }).join('\n')
21
22     if(sortParams !== null){
23         methods = sortToCode(sortParams)
24         classes = makeObjectClass(sortParams)
25         calls = makeMethodsCall(objects,sortParams)
26     }
27
28 classes += removeClassesDuplicates(objects).filter(obj => obj.type
       !== 'PATH').map(element => {
29     return objectToClass(element,sortParams)
30 }).join('\n')
31
32     if(reached){
33         calls += pathToArray(objects)
34         calls += pathToCode(objects)
```

```
35      }
36
37      const code = `public class Main{
38      public static void main (String[] args) {
39      ${variables}
40      ${instances}
41      ${calls}
42      }\n
43  ${methods}
44  }\n
45  ${classes}`;
46      return code;
47  }
```

Listing 4.14: Funzione CanvasInterpreter

Questa funzione svolge le seguenti operazioni:

1. Valida gli oggetti contenuti nell'array di ingresso e genera le variabili associate invocando la funzione *objectToVariables*;
2. Genera le istanze degli oggetti a partire dall'array, utilizzando la funzione *objectToInstance*;
3. Genera le classi associate alle istanze, rimuovendo eventuali duplicati in caso di molteplici oggetti dello stesso tipo, mediante la funzione *objectToClass*;
4. Svolge operazioni secondarie di ordinamento degli oggetti e pulizia;
5. Compone il codice finale andando ad unire le singole sezioni realizzate.

La funzione 4.15 definisce le variabili partendo dagli attributi degli oggetti. Ad esempio, se l'array contiene un oggetto di tipo Box posizionato in coordinate [-2.5, 0.5, 12.5], allora la funzione genererà la seguente variabile "double[] b1_position = -2.5, 0.5, 12.5;" correttamente indentata.

```
1  function objectToVariables(object) {
2      let vars = '';
3      for (const field in object) {
4          if (field !== "id" && field !== "type") {
5              vars = vars + `${variableType[field]} ${object.type
       [0].toLowerCase()}${object.id}_${field} = ${getValue(field,
       object[field])};\n`;
6          }
7      }
8  return vars;
9  }
```

Listing 4.15: Funzione objectToVariables richiamata da CanvasInterpreter

A seguire, la funzione 4.16 restituisce le istanze degli oggetti a partire dall'array. Ad esempio, partendo da un oggetto di tipo Box contenuto nell'array, viene generata la seguente istanza "Box b1 = new Box(b1_position, b1_size);". La funzione *objectToInstanceParams* consente di generare i parametri da inserire all'interno dell'istanza della classe ottenuta.

```
export function objectToInstance(object) {
    return `${capitalize(object.type)} ${object.type[0].
    toLowerCase()}${object.id} = new ${capitalize(object.type)}(${
    objectToInstanceParams(object, object.type)});`;
}
function objectToInstanceParams(object, type) {
let params = '';
let cont = 0;
for (const field in object) {
    if (field !== "id" && field !== "type") {
        if (cont > 0) {
params = params + `,${type[0].toLowerCase()}${object.id}_${field}`
        } else {
params = params + `${type[0].toLowerCase()}${object.id}_${field}`
        }
        cont++;
    }
}
return params;
}
```

Listing 4.16: Funzioni objectToInstance ed objectToInstanceParams richiamate da CanvasInterpreter

Il rendering di elementi grafici è affidato al componente Canvas: basato su R3F e Three.js, consente di visualizzare a schermo quanto contenuto all'interno dell'array di oggetti. All'interno del Canvas sono definite caratteristiche come scena, camera, luci, plane e grids che consentono di visualizzare il mondo 3D visto precedentemente. A seguire, la definizione della struttura del Canvas in R3F.

```
export default function Canvas3f(props) {
let { ... } = props;
const scene = new THREE.Scene();
const cubeTextureLoader = new THREE.CubeTextureLoader();
scene.background = cubeTextureLoader.load([stars, stars, stars,
    stars, stars, stars]);

return (
    <div style={{ minWidth: '300px', maxWidth: '500px', height: '
    500px' }}>
        <Canvas scene={scene} >
            <Suspense fallback={<Loader />}>
```

```
11              {!loading? (<> {FPView ? <PointerLockControls /> : <
    OrbitControls />}
12              <Floor mode={mode} />
13              {(mode === 'ADD' || mode === 'MOVE' || mode === 'PATH'
     || mode === 'RUN') ? <gridHelper args={mode === 'PATH' || mode
     === 'RUN' ? [10, 10] : [20, 20]}></gridHelper> : ''}
14
15              <ambientLight />
16              <pointLight position={[50, 50, 50]} />
17              <PerspectiveCamera position={mode === 'PATH' || mode
    === 'RUN' ? [0, 15, 0] : [-10, 20, 20]} makeDefault />
18
19              <Scene mode={mode} setMode={setMode} selectObj={
    selectObj} setSelectObj={setSelectObj} setAddType={setAddType}
    addType={addType} objects={objects} setObjects={setObjects}
    destinationPath={destinationPath} loading={loading}></Scene>
20                  </>) : null}
21              </Suspense>
22              </Canvas>
23          </div>
24      )
25  }
```

Listing 4.17: Struttura componente Canvas

Il cuore pulsante della visualizzazione degli oggetti è racchiuso nel componente Scene: al suo interno sono implementate varie funzioni, dalle gestioni degli event listeners che consentono la visualizzazione di elementi grafici al click del mouse, alle funzioni che regolano lo spostamento, la modifica e l'eliminazione degli oggetti. Le azioni eseguite variano a seconda della modalità in cui si trova il Canvas, la quale può assumere diversi stati come RUN, PATH, DEL, MOVE ed ADD. Un esempio significativo è rappresentato dal click del mouse, che, a seconda della modalità corrente, permette di:

1. Aggiungere un nuovo oggetto nella posizione specificata dal click;
2. Spostare un oggetto selezionato nella posizione indicata dal click;
3. Definire il percorso da far percorrere all'avatar.

```
1   const handleClick = () => {
2  if (mode === 'ADD' && addType !== null && !overlap) {
       const newSize = defaultValues[`${addType}_size`];
3      const newobj = {
4          id: index,
5          type: addType,
6          position: new THREE.Vector3(highlightPos.x, 0.5,
    highlightPos.z),
```

```
7       size: new THREE.Vector3(newSize[0], newSize[1], newSize[2]),
8           color: 'yellow',
9           animate: true
10      }
11      setObjects([...objects, newobj]);
        setAddType(null);
12      setOverlap(true);
13      setMode(null);                                                  }
14      if (mode === 'MOVE' && selectObj && !overlap) {
        const index = objects.indexOf(selectObj);                   const
        newObj = {
15          id: selectObj.id,
16          type: selectObj.type,
17          position: new THREE.Vector3(highlightPos.x, 0.5,
        highlightPos.z),
18          size: selectObj.size,
19          color: selectObj.color,
20          animate: selectObj.animate
21      }
22      const newObjects = [...objects];
23      newObjects.splice(index, 1, newObj);
        setObjects(newObjects);
        setSelectObj(null);
24      setMode(null);}
25      if (mode === 'PATH' && !overlap) {
26          const newPath = {
27              id: index,
28              type: mode,
29              position: new THREE.Vector3(highlightPos.x, 0,
        highlightPos.z)
30          }
31          var included = false;
32          clickableTiles.forEach(obj => (vectorEquals(obj, newPath.
        position)) ? included = true : '')
33      if (included) {
34              setObjects([...objects, newPath]);
35              setIndex(index + 1);
36      }
37      if (vectorEquals(destinationPath, highlightPos) && included) {
38              setObjects([...objects, newPath]);
39              setMode("RUN")
40      }
41      }
42  }
```

Listing 4.18: Listener onClick all'interno del componente Scene del Canvas

Le modalità PATH e RUN sono utilizzate nel livello 3 e consentono rispettivamente di indicare la fase in cui viene definito il percorso da far svolgere all'avatar e

la fase in cui l'avatar è in movimento e sta raggiungendo il percorso indicato. In figura 4.7 è rappresentata la fase di definizione del path.

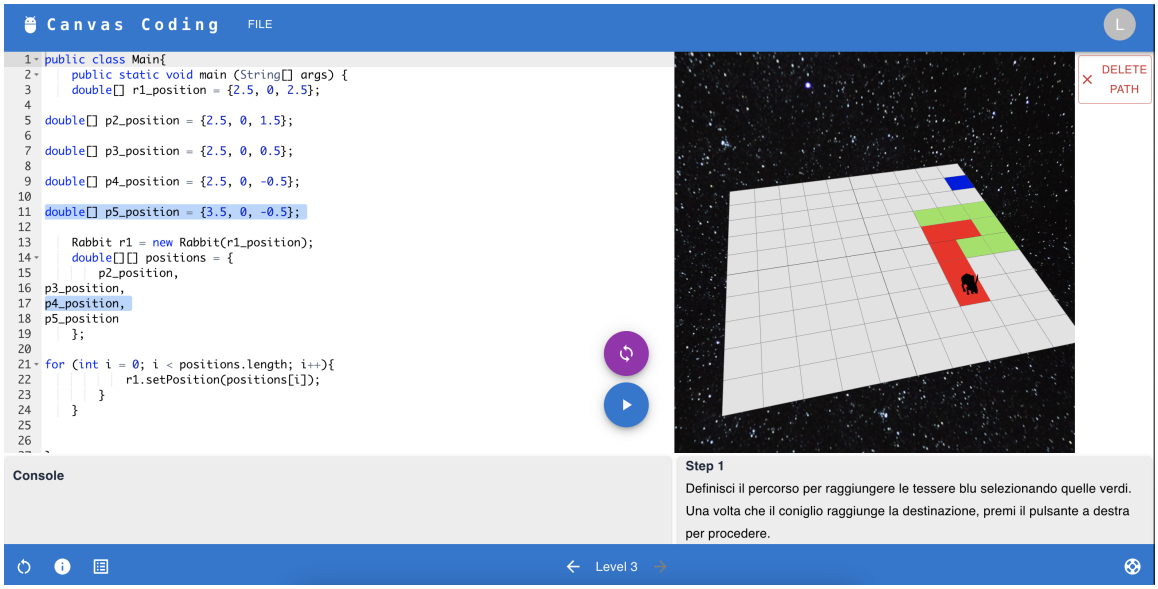

Figura 4.7: Livello 3 del gioco, definizione del path

In figura 4.7, viene evidenziata una funzionalità che consente di facilitare l'utilizzo della piattaforma ai programmatori novizi: ogni volta che un oggetto viene selezionato nel Canvas, questo viene evidenziato nel codice. Inoltre, quando vengono svolte delle modifiche che aggiornano una sola parte del codice, questa viene evidenziata al fine di far comprendere meglio all'utente come un'azione svolta nel Canvas si ripercuote nel codice.

4.2.2 Back-end

Nella realizzazione del server con Spring Boot in Kotlin[8] è stata utilizzata l'architettura **Controller-Service-Repository**. Questa segue il design pattern Model-View-Controller (MVC) ma aggiunge ulteriori strati che consentono di separare le responsabilità e garantire una migliore modularità e manutenibilità del codice, favorendo l'utilizzo dei micro servizi. In figura 4.8 è fornito un esempio del flusso di comunicazione tra i diversi attori di questa architettura.

Questo design pattern è pienamente supportato da Spring Boot, che fornisce all'utente le annotazioni *@Controller*, *@Service* e *@Repository* da aggiungere come marcatori alle singole classi al fine di consentire l'implementazione automatica di

[8]Linguaggio di programmazione.

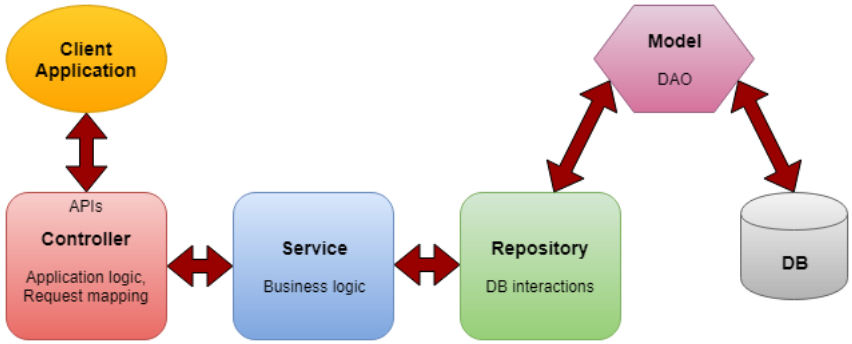

Figura 4.8: Architettura Controller-Service-Repository e flusso di chiamate, fonte `wwww.randikatech.blogspot.com`

una serie di servizi e funzionalità. I componenti alla base di questa architettura sono:

1. **Controller layer**: riceve le richieste HTTP esposte mediante endpoint, le elabora ed invoca i servizi offerti dal Service Layer per ottenere o manipolare dati. I controller sono implementati mediante l'utilizzo delle annotazioni *@Controller* (nel caso in cui restituisce un componente grafico come risposta) *@RestController* (nel caso di interazione con richieste HTTP);

2. **Service layer**: contiene la "business logic" dell'applicazione, in quanto all'interno risiedono le operazioni e le elaborazioni cardine su cui si fonda l'intera piattaforma. Le classi che agiscono da servizi sono precedute dall'annotazione *@Service* di Spring;

3. **Repository layer**: gestisce l'accesso ai dati interagendo con il database o altre fonti. Questo livello ne garantisce un'astrazione, in quanto nasconde i dettagli relativi alla tecnologia di persistenza utilizzata e ai metodi di accesso ai dati. Ad esempio, Spring Data JPA semplifica l'implementazione dei repository, consentendo l'accesso ai dati attraverso chiamate dichiarative.

Considerando i 3 componenti del design pattern adottato, è presentato a titolo di esempio il flusso di chiamate e le implementazioni dei componenti associati agli Step del gioco.

StepController espone gli end-point Rest, contattati dal client attraverso le API messe a disposizione. Ogni end-point è associato ad una funzione contenente:

1. **Mapping URL**: mediante l'utilizzo delle annotazioni @PostMapping, @GetMapping, @PutMapping e @DeleteMapping, è possibile definire il tipo di chiamata REST e l'URL che espone la funzionalità;

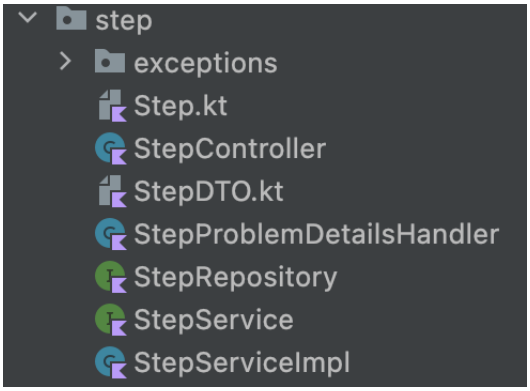

Figura 4.9: File associati alla gestione degli Step del gioco

2. **Request body**: nel caso in cui sia presente un body nella richiesta, questo viene definito attraverso un'astrazione definita DTO [9];

3. **PathVariable**: consente di definire le variabili utilizzate all'interno dell'URL.

```
1  @RestController
2  @CrossOrigin
3  class StepController(
4      private val stepService: StepService
5  ) {
6      @PostMapping("/API/steps")
7      fun postStep(@Valid @RequestBody stepDTO: StepDTO, br :
       BindingResult): StepDTO?{
8          if (br.hasErrors()){
9              val errors = br.allErrors
10             val errMessages = errors.map { it.defaultMessage }
11             throw InvalidCourseDTOException(errMessages)
12         }
13         return stepService.postStep(stepDTO)
14     }
15
16     @GetMapping("/API/step/{id}")
17     fun getStep(@PathVariable id: Int): StepDTO?{
18         return stepService.getStep(id)
19     }
20     ...
21 }
```

Listing 4.19: Implementazione di StepController

[9]Data Transfer Object

Ogni metodo all'interno di StepController invoca una funzione fornita dal **StepService**: questa definisce l'interfaccia della classe, comprendente esclusivamente le firme dei metodi, mentre **StepServiceImpl** presenta le effettive implementazioni. Grazie a Spring Security, è possibile definire un livello di sicurezza per ogni metodo: utilizzando le annotazioni *@Secured*, è possibile specificare il ruolo necessario all'utente per svolgere le funzionalità messe a disposizione dal servizio, insieme ad altri controlli di sicurezza.

```
1  @Service
2  @Transactional
3  class StepServiceImpl(
4      private val stepRepository: StepRepository,
5      private val levelRepository: LevelRepository
6  ) : StepService {
7
8      @Secured("ROLE_TEACHER")
9      override fun putStep(stepDTO: StepDTO): StepDTO? {
10         val newlev = levelRepository.findByIdOrNull(stepDTO.
       level_id)
11
12         if(newlev !== null){
13             val step = stepRepository.findByIdOrNull(stepDTO.id)
       ?: throw StepNotFoundException("Step not found")
14             step.apply {
15                 description = stepDTO.description
16                 action_menu = stepDTO.action_menu
17                 side_menu = stepDTO.side_menu
18                 tip = stepDTO.tip
19                 dialogue = stepDTO.dialogue
20                 level = newlev
21             }
22             return step.toDTO()
23         }else{
24             throw LevelNotFoundException("Level not found!")
25         }
26     }
27     override fun getStepsByLevelId(levelId: Int): List<StepDTO> {
28         val level = levelRepository.findByIdOrNull(levelId)
29         if(level != null){
30             return stepRepository.findStepsByLevel_IdOrderByIdAsc(
       levelId).map { it.toDTO() }
31
32         }else{
33             throw LevelNotFoundException("Level not found!")
34         }
35     }
36 }
```

Listing 4.20: Implementazione di StepServiceImpl

All'interno di questa classe, sono inoltre richiamati i metodi di **StepRepository**, che consentono di accedere ai dati contenuti nel database e di effettuare modifiche. In particolare, grazie all'utilizzo di JPA, l'accesso ai dati non richiede la definizione di query esplicite, a meno che non si tratti di operazioni più complesse. Questa semplicità è resa possibile anche dall'utilizzo della classe Step, che è annotata con *@Entity* e viene mappata direttamente con i dati nel database. In questo modo, Spring Data JPA genera automaticamente le query di accesso ai dati e verifica la coerenza dei dati nel database rispetto alla struttura definita nella classe, ad ogni avvio dell'applicazione.

```
1  @Entity
2  class Step(
3      var action_menu: String?,
4      var description : String?,
5      var dialogue: String?,
6      var side_menu: String?,
7      var number: Int?,
8      var tip: String?,
9      @ManyToOne
10     var level : Level
11     ): EntityBase<Int>()
12 fun StepDTO.toEntity(level: Level): Step{
13     return Step(action_menu,description,dialogue, side_menu,
    number, tip, level = level)}
```

Listing 4.21: Implementazione della entity Step

I dati memorizzati all'interno del database relazionale offerto da postgreSQL, seguono lo schema in figura 4.10.

Questo contiene le seguenti tabelle:

- **Profile**: rappresenta il profilo degli utenti, non contiene la password in quanto memorizzata e gestita da Keycloak. In fase di creazione delle utenze, queste vengono create sia su Keycloak che su postgreSQL, anche se hanno scopi differenti;
- **Course**: corrisponde alla classe frequentata dagli studenti, viene generata dai docenti. L'ID del corso è richiesto in fase di creazione dell'utenza;
- **GameSession**: rappresenta una sessione di gioco dell'utente per un determinato step del livello. Essendo univoca per ogni step, consente all'utente di muoversi attraverso i livelli senza perdere il codice generato. Inoltre, viene utilizzato per tenere traccia dei progressi dell'utente nel gioco;
- **Step**: contiene le informazioni di visualizzazione da inserire nell'Action Menu e Side Menu, insieme ai controlli da svolgere al fine di verificare il completamento dello step;

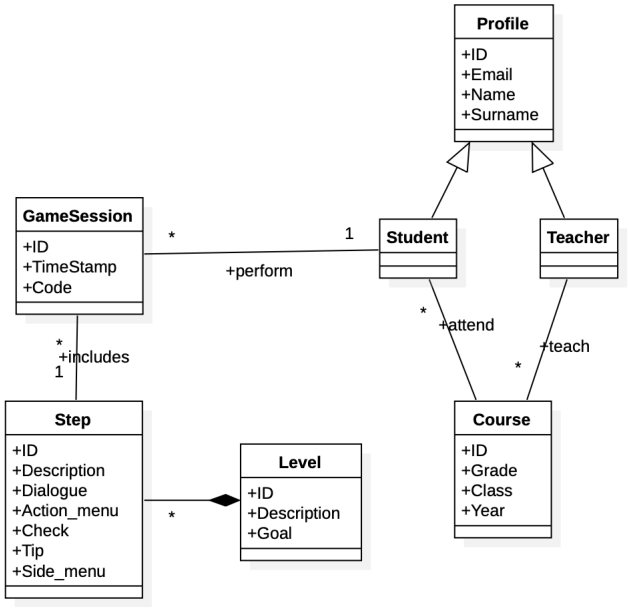

Figura 4.10: Schema UML del database

- **Level**: rappresenta informazioni associate al livello, come descrizione ed obiettivi, mostrati a schermo nell'interfaccia accessibile dal footer.

Capitolo 5

Valutazione sperimentale

5.1 Introduzione

Al fine di supportare lo sviluppo del gioco Canvas Coding e di avere un'idea più completa della buona riuscita del lavoro svolto, è stato condotto un cosiddetto Usability Testing o test d'usabilità.

Questa tipologia di test valuta quanto bene un'applicazione è stata progettata per soddisfare la facilità d'uso per uno specifico gruppo demografico di utenze. Durante la valutazione, è possibile osservare come le persone utilizzano le interfacce al fine di individuarne i punti di forza e di debolezza. L'obiettivo principale del test è dunque quello di verificare l'usabilità e la funzionalità di un sistema interattivo tramite diversi criteri, per poter correggere le possibili problematiche riscontrate prima di un rilascio pubblico.

I test di usabilità sono in genere composti da 3 fasi distinte:

- **Pianificazione**: durante questa fase, vengono prese le decisioni principali riguardo i test da condurre, come ad esempio il numero ed il target di partecipanti, suddivisione dei ruoli, strumenti da utilizzare durante lo svolgimento dei test, metriche da adottare per la valutazione dei risultati, composizione e scelta dei questionari da utilizzare, stesura dello script;
- **Esecuzione**: corrisponde alla fase di test vera e propria, condotta sui singoli tester presi in analisi. Durante lo svolgimento del test, vengono raccolti i dati necessari alla fase di analisi successiva utilizzando gli strumenti definiti nella fase precedente;
- **Analisi dei risultati ottenuti**: sulla base dei dati raccolti durante la fase di esecuzione, tenendo in considerazione le metriche definite nella fase di pianificazione e i risultati dei questionari, è possibile comprendere i punti di forza e di debolezza dell'oggetto esaminato.

Seguendo questa suddivisione, le sezioni a seguire presentano i test di usabilità condotti e i risultati ottenuti.

5.2 Pianificazione

Durante questa fase vengono definite le modalità di esecuzione del test e le caratteristiche principali, al fine di ottenere dei risultati veritieri che possano risultare utili a correggere le eventuali problematiche individuate.

5.2.1 Partecipanti, ruoli e strumenti

Il primo step della fase di pianificazione consiste nella definizione del numero di partecipanti e delle loro caratteristiche. Secondo quanto definito nelle linee guida di Nielsen, il numero di partecipanti adatto a trovare il maggior quantitativo di problemi è 5: utilizzare più tester diventa contro indicativo, in quanto causa solo un aumento dei costi e non dell'affidabilità dei risultati ottenuti. Per questo motivo, si è deciso di scegliere 5 partecipanti da sottoporre al test: questi sono stati scelti per adattarsi, nel miglior modo possibile, al tipo di utenza a cui il gioco Canvas Coding è destinato. In questo caso, i soggetti del test devono possedere le seguenti caratteristiche:

- Studenti o studentesse di istituti superiori di tipo tecnico o scientifico ad indirizzo informatico;
- Età compresa tra i 16 ed i 19 anni;
- Conoscenza di base dei i concetti relativi alla programmazione ad oggetti e procedurale.

Successivamente, è stato definito un team composto da due membri che ricoprono rispettivamente i ruoli di facilitatore ed osservatore. Questi ruoli hanno compiti ed obiettivi differenti:

- facilitatore: ha il compito di condurre il test seguendo alla lettera quanto definito nello script, interagendo il meno possibile con il partecipante al fine di evitare condizionamenti o bias [1]. Proprio per questo motivo, tutti coloro che hanno preso parte alla realizzazione dell'oggetto da testare non devono ricoprire il ruolo di facilitatore, in quanto potrebbero involontariamente condizionare i tester;

[1]Particolari euristiche usate per generare opinioni o esprimere dei giudizi, su cose di cui non si è mai avuto esperienza diretta

- osservatore: il ruolo dell'osservatore è quello di raccogliere dati, prendere appunti, scattare foto e video, senza interagire con il soggetto del test. In questo caso, lo sviluppatore può ricoprire questo ruolo poiché le interazioni con il partecipante sono fortemente limitate.

A seguire, sono stati definiti gli strumenti hardware e software necessari per eseguire con successo i test, pianificati per essere svolti online. In questo caso, l'equipaggiamento include:

- 2 computer, uno per il partecipante e l'altro per il facilitatore;
- Applicativo di teleconferenza installato e configurato in entrambi i computer;
- Connessione ad internet per entrambi i computer;
- Software di registrazione schermo per il computer del facilitatore;
- Fotocamera per svolgere foto;
- Microfono per registrare audio.

Per evitare di installare l'intero progetto sui computer dei tester o di rendere pubblico il gioco ed accessibile attraverso internet, si è scelto di utilizzare la condivisione dello schermo e cedere il controllo del computer utilizzato dal facilitatore, in quanto contenente l'intero progetto funzionante e configurato: in questo modo, i partecipanti utilizzano i loro computer solo come veicolo per accedere a quello del facilitatore, con cui svolgeranno l'intero test.

5.2.2 Task e metodologie

Una volta definiti i partecipanti, i ruoli e gli strumenti da utilizzare, sono stati definiti i task da sottoporre: ognuno ricopre diversi aspetti e funzionalità del gioco, e complessivamente consentono di testare il progetto nella sua interezza.

Per ognuno di questi task sono definiti:

- **Nome**: riassume il contenuto del task e consente di identificarlo;
- **Scenario**: corrisponde ad una contestualizzazione fornita dal facilitatore durante la somministrazione del test, ha l'obiettivo di far sentire il soggetto parte del sistema, senza influenzarne i risultati. Ad esempio, invece di fornire istruzioni del tipo "Seleziona un oggetto dal Side menu e modifica l'attributo colore", viene fornito al partecipante qualcosa del tipo "Sei appena tornato a casa da un corso di informatica sulle classi e gli oggetti, ma non hai capito molto. Vuoi evitare di utilizzare il libro per rivedere questi concetti, quindi utilizzi il gioco Canvas Coding per creare un oggetto e personalizzarlo";

- **Obiettivo**: indica gli obiettivi associati al completamento del task in termini di funzionalità utilizzate con successo;
- **Criteri**: per ogni task, sono definiti dei criteri di completamento in percentuale che indicano quanto gli obiettivi sono stati raggiunti o meno dall'utente nella fase di test;
- **Metodologia**: indica la metodologia di conduzione del test adottata. In questo caso, tutti i task sono stati svolti utilizzando il *Think Aloud*. Secondo questa metodologia, il tester deve esprimere ad alta voce tutti i pensieri che gli passano per la mente mentre agisce sull'oggetto da testare: in questo modo, è più facile analizzare l'intero flusso di pensiero dell'utente mentre utilizza il gioco. Esistono anche altre metodologie come il *Cooperative Evaluation*, che prevede l'affiancamento del facilitatore nell'esecuzione dei task.

Tutte le sessioni di test sono state svolte online. A seguire sono presentati i 7 task sviluppati.

N	**Nome**	**Scenario**	**Obiettivi**	**Criteri**
T1	Definizione di un oggetto e completamento dei primi step del livello 1	Sei appena tornato a casa da un corso di informatica sulle classi e gli oggetti, di cui non hai capito molto. Vuoi evitare di utilizzare il libro per rivedere questi concetti, e per questo motivo utilizzi il gioco Canvas Coding per apprendere in modo alternativo	Definire un oggetto utilizzando il Canvas; Completa i primi 2 passaggi; Utilizza i collegamenti tra Canvas e codice;	100% Definizione di un oggetto E completa i passaggi E utilizza i collegamenti tra Canvas e codice; 80% Definizione di un oggetto E Completa i passaggi; 50% Definisce un oggetto; 0% Oggetto non definito;

T2	Completa livello 1	Sei uno studente curioso e determinato, per questo prima di completare il livello 1, decidi di esplorare tutte le potenzialità della piattaforma	Completa il livello 1; Apporta modifiche al codice; Sincronizza il Canvas con il codice; Compilare il codice;	100% Scrivi codice E Sincronizza Canvas con codice E compila E Completa il livello 1; 80% Modifica codice E (Sincronizza tela con codice O compila) E Completa il livello 1; 60% Sincronizza il canvas con il codice/compila E Completa il livello 1; 40% Modifica codice E (sincronizza Canvas con codice/compila); 0% Livello 1 non completato E Codice non modificato E Non compila/sincronizza;
T3	Navigazione tra livelli	Non hai capito alcuni passaggi del primo livello, quindi decidi di ripeterli e utilizzare i suggerimenti forniti dall'interfaccia fino a raggiungere il secondo livello	Torna al livello precedente; Aprire e utilizzare i suggerimenti; Completa il livello 1;	100% Scrivi codice E Sincronizza Canvas E compila E Completa il livello 1; 80% Modifica codice E (Sincronizza Canvas con codice O compila) E Completa il livello 1; 60% Sincronizza il Canvas con il codice O compila E Completa il livello 1; 40% Modifica codice E (sincronizza Canvas con codice O compila); 0% Livello 1 non completato E Codice non modificato E Non compila O Sincronizza;

T4	Utilizza funzioni nel Side Menu	Sei a scuola, hai appena seguito una lezione sugli algoritmi di ordinamento, ma non hai capito bene le differenze. Decidi di utilizzare le funzioni di ordinamento disponibili sulla piattaforma, eliminare gli oggetti che ritieni ridondanti e spostarne altri.	Utilizza il Action Menu; Elimina almeno un oggetto tramite Canvas; Sposta almeno un oggetto tramite Canvas; Eseguire almeno un ordinamento; Segue le indicazioni del livello;	100% Utilizza Action Menu E Elimina un oggetto E Sposta un oggetto E Esegui ordinamento E Segui i passaggi del livello; 80% Utilizza il Action Menu E Elimina un oggetto E Sposta un oggetto E Esegui l'ordinamento; 80% Utilizza Action Menu E (Elimina un oggetto OPPURE Sposta un oggetto) E Esegui l'ordinamento E Segui i passaggi; 60% Utilizza Action Menu E (Elimina un oggetto OPPURE Sposta un oggetto) E Esegui ordinamento; 60% Utilizza Action Menu E Esegui ordinamento E Segui i passaggi; 40% Utilizza Action Menu E (Elimina un oggetto OPPURE Sposta un oggetto); 40% Utilizza Action Menu ED Esegui l'ordinamento; 0% Non utilizza Action Menu;

T5	Ordina gli oggetti ed esporta il codice	Domani hai un esame di informatica sugli algoritmi di ordinamento e vorresti generare alcuni esempi funzionanti di codice. Decidi quindi di utilizzare la piattaforma per generare tutti i possibili algoritmi di ordinamento e di esportare il codice per ciascun algoritmo creato.	Genera tutti i tipi disponibili di algoritmi di ordinamento; Esporta il codice;	100% Utilizza tutti gli algoritmi di ordinamento disponibili E esporta il codice per ognuno di essi; 80% Utilizza tutti gli algoritmi di ordinamento disponibili ED esporta il codice almeno 1 volta; 60% Utilizza tutti gli algoritmi di ordinamento disponibili; 60% Non utilizza tutti gli algoritmi di ordinamento E esporta il codice almeno 1 volta; 40% Non utilizza tutti gli algoritmi di ordinamento; 20% Esporta il codice; 0% Non utilizzare alcun algoritmo di ordinamento E non esporta il codice;
T6	Definisci e risolvi il percorso utilizzando iterazioni e vettori	Ti è stato assegnato un compito a casa che richiede l'uso di strutture e vettori. Decidi di utilizzare la piattaforma per visualizzare graficamente queste nozioni.	Definire un vettore di oggetti utilizzando il Canvas; Definire una struttura iterativa; Definire un percorso; Risolvi il percorso;	100% Definisci un percorso E Definisci un vettore E Utilizza struttura iterativa E Risolvi il percorso; 60% Definisci un percorso E Definisci un vettore E Usa il ciclo for; 40% Definisci percorso E (Definisci vettore O usa per loop); 0% Non definire un percorso;

T7	Definisci un percorso crea gli ostacoli	Sei diventato un utente esperto della piattaforma e decidi di unire tutte le nozioni apprese, in termini di definizione di oggetti e posizionamento. Decidi quindi di complicare il percorso aggiungendo ostacoli, scrivendo il codice richiesto.	Definire almeno un oggetto utilizzando il codice; Risolvi il percorso utilizzando il codice o il Canvas;	100% Definisci almeno un oggetto utilizzando il codice E Risolvi il percorso; 80% Definire almeno un oggetto utilizzando il codice; 50% Risolvi il percorso; 0% Non definisce un oggetto utilizzando il codice E non risolve il percorso;

Tabella 5.1: Task sottoposti nella fase di valutazione

5.2.3 Metriche

Al fine di valutare l'andamento complessivo del test, è necessario definire delle metriche di valutazione. Quelle adottate sono:

- **Tempo**: corrisponde al tempo impiegato in secondi dal tester per completare il task;
- **Completamento del task in percentuale**: basandosi sui criteri definiti in tabella 5.1, un task viene completato con successo quando il partecipante indica di aver trovato la risposta o raggiunto l'obiettivo;
- **Errori critici**: numero intero che rappresenta le deviazioni al completamento dagli obiettivi dell'attività definiti in tabella 5.1, in modo che il partecipante non possa completarla. Il partecipante può essere consapevole o meno che l'obiettivo dell'attività non è corretto o è incompleto;
- **Misure soggettive**: valutazioni definite dai partecipanti riguardo la soddisfazione, facilità d'uso, facilità di trovare informazioni, sperimentate durante l'utilizzo del gioco. Questa metrica viene raccolta subito dopo aver terminato il test, mediante la somministrazione di un questionario apposito;

- **Raccomandazioni**: il partecipante, sotto forma un colloquio informale alla fine del test, indica cosa ha gradito di più, cosa meno, ed eventuali consigli e migliorie. Questa fase prende anche il nome di "debriefing".

5.2.4 Domande post-test

Per raccogliere i dati soggettivi sull'usabilità del gioco, è stato utilizzato un questionario denominato "SUS" (System Usability Scale), somministrato alla fine del test. Inventato da Jhon Brooke nel 1986, è un sistema semplice ma affidabile che consente di misurare l'usabilità percepita di un sistema. E' composto da 10 domande, ognuna con 5 possibili risposte, e mediante l'applicazione di una formula consente di produrre un punteggio che va da 0 a 100. La soglia minima da raggiungere per ottenere un punteggio sopra la media è di 68.

A seguire, sono presentate le domande inserite all'interno del questionario:

1. Sono dell'idea che userei questo sistema frequentemente;
2. Ho trovato il sistema inutilmente complesso;
3. Sono dell'idea che questo sistema sia facile da usare;
4. Sono dell'idea di aver necessità di supporto tecnico per essere in grado di usare questo sistema;
5. Sono dell'idea che le varie funzioni nel sistema siano ben integrate;
6. Sono dell'idea che nel sistema ci siano troppe inconsistenze;
7. Riuscirei facilmente a immaginare che molte persone sarebbero in grado di imparare facilmente come usare questo sistema;
8. Ho trovato il sistema molto macchinoso;
9. Mi sono sentito a mio agio nell'utilizzare il sistema;
10. Avevo bisogno di apprendere molte informazioni prima di potermi sentire a mio agio con il sistema.

5.3 Esecuzione

In questa sezione, sono presentati i risultati dell'esecuzione dei test: per ogni partecipante e per ogni task sono indicate le metriche valutate, associate ai valori ottenuti. Inoltre, sono definiti i ruoli di facilitatore, osservatore e delle note aggiuntive.

5.3.1 Intervista 1

Il primo soggetto intervistato si chiama Gianvito, uno studente di 18 anni frequentante il 4 anno in un istituto tecnico superiore in provincia di Bari.

- Facilitatore: Simone D'Ingianna
- Osservatore: Antonello Caputo

N Task	Metrica	Valori
1	Tempo	136.5 secondi
	Completamento in %	100%
	Errori critici	0 *
2	Tempo	139.5 secondi
	Completamento in %	80%
	Errori critici	1 (Non sincronizza)
3	Tempo	97.5 secondi
	Completamento in %	60%
	Errori critici	1 (Non usa i suggerimenti)
4	Tempo	133 secondi
	Completamento in %	60% *
	Errori critici	1 (Non cancella o sposta gli oggetti)
5	Tempo	76.5 secondi
	Completamento in %	80% **
	Errori critici	1 (Esporta il codice una sola volta)
6	Tempo	77 secondi
	Completamento in %	100%
	Errori critici	0
7	Tempo	368 secondi
	Completamento in %	50%
	Errori critici	1 (Non riesce a definire l'ostacolo)

Tabella 5.2: Risultati e metriche intervista 1

Note: *[2], **[3].
Nella fase di debriefing finale, è emerso che:

- **Cosa ha gradito di più**: "Finalmente ho una interfaccia grafica dove posso vedere cosa sto facendo"

[2]Richiede la traduzione delle indicazioni

[3]Tempo impattato da problema al click del mouse con TeamViewer

- **Cosa ha apprezzato meno**: “Nulla”
- **Suggerimenti**: “Aggiungerei traduzioni ed altri linguaggi di programmazione”.

5.3.2 Intervista 2

Il secondo soggetto intervistato si chiama Vincenzo, uno studente di 17 anni frequentante il 4 anno in un liceo scientifico ad indirizzo informatico in provincia di Messina.

- Facilitatore: Simone D’Ingianna
- Osservatore: Antonello Caputo

N Task	Metrica	Valori
1	Tempo	127 secondi
	Completamento in %	100%
	Errori critici	0
2	Tempo	180 secondi
	Completamento in %	100%
	Errori critici	0
3	Tempo	178 secondi
	Completamento in %	60%
	Errori critici	1 (Non trova i suggerimenti)
4	Tempo	345 secondi
	Completamento in %	60%
	Errori critici	1 (Non si muove o cancella oggetti)
5	Tempo	136 secondi
	Completamento in %	60%
	Errori critici	1 (Non esporta il codice)
6	Tempo	132 secondi *
	Completamento in %	100%
	Errori critici	0
7	Tempo	457 secondi
	Completamento in %	100%
	Errori critici	0

Tabella 5.3: Risultati e metriche intervista 2

Note: *[4]

[4]Richiede la traduzione delle indicazioni

Nella fase di debriefing finale, è emerso che:

- **Cosa ha gradito di più**: "L' interfaccia è molto pulita, ho apprezzato i suggerimenti e l'aiuto visivo. E' intuitiva ma io mi sentivo in difficoltà in quanto non conosco bene il codice.. ma la piattaforma è molto usabile"
- **Cosa ha apprezzato meno**: "Interfaccia in inglese spesso mi confondeva"
- **Suggerimenti**: "Qualche suggerimento in più e l'accesso agli esempi in modo più semplice".

5.3.3 Intervista 3

Il terzo soggetto intervistato si chiama Francesco, uno studente di 18 anni frequentante il 5 anno in un istituto tecnico superiore in provincia di Torino.

- Facilitatore: Simone D'Ingianna
- Osservatore: Antonello Caputo

N Task	Metrica	Valori
1	Tempo	180 seconds
	Completamento in %	100%
	Errori critici	0
2	Tempo	110 secondi
	Completamento in %	40%
	Errori critici	1 (Non completa il livello)
3	Tempo	180 secondi
	Completamento in %	60%
	Errori critici	1 (Non usa i suggerimenti)
4	Tempo	176 secondi
	Completamento in %	60%
	Errori critici	2 (Non sposta e cancella oggetti)
5	Tempo	137 secondi
	Completamento in %	100%
	Errori critici	0
6	Tempo	71 secondi
	Completamento in %	100%
	Errori critici	0
7	Tempo	259 secondi
	Completamento in %	100%
	Errori critici	0

Tabella 5.4: Risultati e metriche intervista 3

Nella fase di debriefing finale, il tester ha affermato che:

- **Cosa ha gradito di più**: "Il gioco finale con il coniglio dove superare gli ostacoli è molto carino, anche la creazione della sfera e del cubo anche ma preferisco l'ultimo livello"
- **Cosa ha apprezzato meno**: "Nulla"
- **Suggerimenti**: "L'interfaccia è semplice ma fa il suo lavoro, aggiungerei degli esercizi in più...".

5.3.4 Intervista 4

Il quarto soggetto intervistato si chiama Francesco, uno studente di 18 anni frequentante il 5 anno in un istituto tecnico superiore in provincia di Bari.

- Facilitatore: Simone D'Ingianna
- Osservatore: Antonello Caputo

N Task	Metrica	Valori
1	Tempo	68 secondi
	Completamento in %	100%
	Errori critici	0
2	Tempo	69 secondi
	Completamento in %	80%
	Errori critici	1 (Non sincronizza)
3	Tempo	99 secondi
	Completamento in %	60%
	Errori critici	1 (Non usa i suggerimenti)
4	Tempo	135 secondi
	Completamento in %	100%
	Errori critici	0
5	Tempo	77 secondi
	Completamento in %	100%
	Errori critici	0
6	Tempo	45 secondi
	Completamento in %	100%

	Errori critici	0
7	Tempo	405 secondi
	Completamento in %	50%
	Errori critici	1 (Non definisce ostacolo)

Tabella 5.5: Risultati e metriche intervista 4

Nella fase di debriefing finale, il tester ha affermato che:

- **Cosa ha gradito di più**: "La grafica non è male, aggiungere un oggetto graficamente e poterlo modificare e vedere la modifica in tempo reale mi è piaciuto molto.. Anche la funzione sort l'ho trovata utile"
- **Cosa ha apprezzato meno**: "Non sono riuscito a capire come aggiungere un altro oggetto tramite codice nell'ultimo livello... Per questo mi è piaciuto meno"
- **Suggerimenti**: "Implementare altre funzioni oltre l'ordinamento oppure di utilizzare cicli e strutture, ad esempio for e while che si vedano in tempo reale nella parte grafica...".

5.3.5 Intervista 5

Il quinto soggetto intervistato si chiama Flavia, una studentessa di 17 anni frequentante il 4 anno in un liceo scientifico ad indirizzo informatico in provincia di Bari.

- Facilitatore: Simone D'Ingianna
- Osservatore: Antonello Caputo

N Task	**Metrica**	**Valori**
1	Tempo	135 secondi
	Completamento in %	100%
	Errori critici	0
2	Tempo	108 secondi
	Completamento in %	100%
	Errori critici	0
3	Tempo	89 secondi
	Completamento in %	60%
	Errori critici	1 (Non trova i suggerimenti)

4	Tempo	221 secondi
	Completamento in %	100%
	Errori critici	0
5	Tempo	165 secondi
	Completamento in %	80%
	Errori critici	1 (Genera il codice una sola volta)
6	Tempo	60 secondi
	Completamento in %	100%
	Errori critici	0
7	Tempo	630 secondi
	Completamento in %	50%
	Errori critici	1 (Non definisce ostacolo)

Tabella 5.6: Risultati e metriche intervista 5

Nella fase di debriefing finale, è emerso che:

- **Cosa ha gradito di più**: "La piattaforma è molto intuitiva, anche se non si ha una base solida di informatica, è intuitivo svolgere le operazioni..."
- **Cosa ha apprezzato meno**: "Ultimo esercizio in quanto non sono riuscita a comprenderlo a pieno"
- **Suggerimenti**: "Aggiungerei qualche suggerimento in più, anche tecnico o teorico ".

5.4 Risultati

Dalle sessioni di test di usabilità è stato possibile ricevere informazioni e feedback interessanti riguardo il gioco sviluppato. Nel complesso, anche se ci sono stati alcuni casi in cui i partecipanti erano chiaramente confusi sulle azioni da intraprendere per completare i task previsti, la sessione di test ha prodotto dei risultati molto soddisfacenti.

I risultati del questionario di fine test (SUS) riportati in tabella 5.7 sono stati abbastanza positivi: in media si è ottenuto un punteggio di 77 punti, ben al di sopra della soglia media di 68 punti.

Tester	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	RAW	FINAL
Gianvito	4	1	3	1	2	1	1	1	1	5	22	55
Vincenzo	5	1	4	2	5	1	5	1	5	2	37	92.5
Francesco	3	1	5	1	5	2	5	1	5	2	36	90

Tester	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	RAW	FINAL
Francesco	3	1	4	2	3	2	4	2	4	1	30	75
Flavia	4	2	4	2	3	2	5	2	4	3	29	72.5

Tabella 5.7: Risultati Post-test SUS

Come è possibile notare, il punteggio ricevuto dal primo partecipante si discosta molto dagli altri: molteplici possono esserne le cause, come ad esempio una possibile incomprensione delle domande scritte nel test che hanno generato risposte contrastanti tra loro, ancor di più se confrontate con le risposte aperte fornite nella fase di debriefing.

Ad esempio, la risposta fornita alla domanda 7 "Riuscirei facilmente a immaginare che molte persone sarebbero in grado di imparare facilmente come usare questo sistema" è stata "In completo disaccordo", anche se la risposta alla prima domanda "Sono dell'idea che userei questo sistema frequentemente" la risposta è stata molto affermativa, cosi come nella fase di debriefing ha affermato di riuscire finalmente a visualizzare graficamente i concetti teorici. Lo stesso andamento incoerente lo si ritrova per la risposta alla domanda 9.

Sicuramente, una problematica comune ai primi 2 partecipanti è stata l'assenza della lingua italiana nel gioco: questo ha necessitato l'ausilio del facilitatore nel fornire una traduzione dei comandi mostrati, dopo che il secondo partecipante ha affermato *"Qua ci vuole la traduzione in italiano, io ho problemi con l'inglese"*. Nonostante questo però, entrambi i partecipanti hanno dichiarato di aver trovato il gioco facile da utilizzare e non tedioso. Per questo motivo, dopo i primi 2 test, si è deciso di tradurre in Italiano tutti i dialoghi presenti nel gioco, al fine di ottenere dei risultati più veritieri e non influenzati da questo fattore di incomprensione.

Task	**Valore max (s)**	**Valore min (s)**	**Valore medio (s)**
1	180 (P3)	68 (P4)	127.7
2	180 (P2)	69 (P4)	121.9
3	180 (P3)	89 (P5)	128.9
4	345 (P2)	133 (P1)	190
5	165 (P5)	76.5 (P1)	118.1
6	132 (P2)	45 (P4)	77
7	630 (P5)	259 (P4)	423.8

Tabella 5.8: Tempo trascorso per task medio

In tabella 5.8 è mostrato il valor medio del tempo impiegato per il completamento di ciascun task. Al fine di favorire la creatività dei partecipanti durante l'utilizzo del gioco, si è deciso di non impostare un limite massimo a questa metrica, e di

interrompere il cronometro solo qualora l'utente avesse completato con successo tutti gli obiettivi del task o si fosse arreso volontariamente.

Da questa tabella è possibile quindi comprendere come il tempo trascorso su ogni task sia fortemente influenzato dall'indole personale dell'utente: i partecipanti 4 e 5 si sono confermati i più veloci, mentre il 2 e 3 i più riflessivi e cauti. A prescindere dalla loro indole, il tempo trascorso ci conferma che task più creativi e stimolanti, come ad esempio il 4 e 7, sono in grado di tenere vivo l'interesse degli studenti nel completamento del livello.

Grazie all'utilizzo della metodologia *Think Aloud*, sono emerse diverse criticità del gioco:

- Durante l'esecuzione del task 7, il primo partecipante dichiara *"sono perplesso, non sono convinto di poter riuscire a definire un nuovo oggetto. Non ho capito cosa devo fare"*. Questo trend viene confermato dal quarto partecipante, che afferma *"Non riesco a capire come aggiungere l'oggetto"*. Una soluzione potrebbe essere quella di lasciare la possibilità di inserire gli oggetti graficamente anche nel 3 livello, funzionalità inibita per motivare gli utenti a ricordare i comandi da utilizzare e ricorrere al solo codice;

- Il primo partecipante afferma durante l'esecuzione del task 6 "Ok, non capisco a cosa può servire questa attività", anche se questa affermazione stride molto con quanto affermato dal terzo partecipante, che trova invece il livello 3 molto stimolante ed interattivo.

Task	**Goal**	**Tester 1**	**Tester 2**	**Tester 3**	**Tester 4**	**Tester 5**	**%**
1	1	X	X	X	X	X	100
	2	X	X	X	X	X	100
	3	X	X	X	X	X	100
2	1	X	X		X	X	80
	2	X	X	X		X	80
	3		X	X	X	X	80
	4	X	X	X	X	X	100
3	1	X	X	X	X	X	100
	2						0
	3	X	X	X	X	X	100
4	1	X	X	X	X	X	100
	2				X	X	40
	3				X	X	40
	4	X	X	X	X	X	100
	5	X	X	X	X	X	100
5	1	X	X	X	X	X	100

	2	X		X	X	X	80
6	1	X	X	X	X	X	100
	2	X	X	X	X	X	100
	3	X	X	X	X	X	100
	4	X	X	X	X	X	100
7	1		X	X			40
	2	X	X	X	X	X	100

Tabella 5.9: Risultati complessivi metrica completamento in percentuale

Inoltre, analizzando nel complesso la metrica relativa al completamento in percentuale degli obiettivi dei singoli task in tabella 5.9, è possibile trarre diverse conclusioni:

- Nessuno studente è riuscito a completare il secondo obiettivo previsto dal task 3, ovvero individuare ed utilizzare i suggerimenti forniti dal gioco. Questo evidenzia come, sebbene la funzionalità sia presente, risulti difficile da individuare e per questo non venga utilizzata, anche se esplicitamente richiesta da alcuni partecipanti;
- I primi 3 partecipanti non sono riusciti ad utilizzare le funzionalità di spostamento ed eliminazione degli oggetti, rappresentate dagli obiettivi 2 e 3 del task 4. Anche in questo caso, seppur meno grave, dimostra come sia necessario fornire dei metodi alternativi di accesso a tali strumenti.
- Un errore critico molto comune consiste nella difficoltà nel definire un ostacolo mediante l'utilizzo del solo codice, come evidenziato dal primo obiettivo del task 7. Questo risultato evidenzia come non tutti gli studenti siano in grado di apprendere allo stesso modo le nozioni, e che dunque sia necessario facilitare anche coloro che non riescono, mediante l'inserimento di suggerimenti ad hoc oppure non inibendo la possibilità di inserire oggetti con il canvas.

Capitolo 6

Conclusioni

Il progetto di tesi è stato concepito con l'obiettivo di realizzare uno strumento che utilizzasse in modo innovativo la programmazione creativa al fine di risolvere le principali sfide sperimentate durante i corsi introduttivi alla programmazione.

Durante la fase preliminare è stata condotta una ricerca per individuare, da una parte, le difficoltà e le limitazioni sperimentate dagli studenti durante i corsi di informatica, dall'altra i fattori che ne influenzano positivamente l'apprendimento. A partire dalle prime osservazioni, è emerso che le arti visive e la creatività possano risultare potenti mezzi per motivare ed attrarre l'interesse degli alunni, evitando una percezione complessa, tediosa ed antisociale della disciplina.

Successivamente, è stata analizzata l'applicazione della creatività all'insegnamento dell'informatica, focalizzando l'attenzione sul Creative Coding e sugli strumenti esistenti che lo adoperano. A conclusione della fase di ricerca sono state esaminate ulteriormente le limitazioni individuate, quali la difficoltà nell'integrare le attività creative nei curricula scolastici, la carenza di strumenti efficienti per individuare gli errori di programmazione, l'assenza di mezzi di visualizzazione grafica a supporto degli studenti e la necessità di formare gli insegnanti all'utilizzo di nuove piattaforme e metodologie didattiche.

Con l'obiettivo di superare queste limitazioni, unitamente a tutti i vantaggi dell'utilizzo della programmazione creativa nel contesto educativo, sono state proposte 3 soluzioni differenti, ognuna con i propri punti di forza e debolezza. Dopo un'analisi approfondita delle tre proposte, si è scelto di proseguire con la progettazione ed implementazione della piattaforma di gioco Canvas Coding, essendo apparsa come l'opzione migliore in grado di superare tutte le limitazioni individuate. Essa si contraddistingue per l'utilizzo del gamification, come strumento per rafforzare la sfera motivazionale degli studenti, e del Creative Coding che sfrutta un approccio divertente e coinvolgente, al fine di conseguire risultati di apprendimento ottimali.

La struttura di base del gioco è stata realizzata tenendo in considerazione da una parte i componenti principali degli ambienti di sviluppo e dall'altra i giochi interattivi multimediali. Per consentire all'utente di scrivere ed apprendere la sintassi alla maniera "tradizionale", sono stati utilizzati un Editor di codice, una Console per visualizzare l'esito della compilazione ed un Explorer dei file del progetto. Tra gli elementi tipici dei giochi multimediali, invece, sono stati ereditati il Side Menu ed una vignetta per la rappresentazione delle indicazioni del gioco, in modo da favorire l'interazione con l'utente. Infine, ispirandosi al paradigma del "Direct Manipulation", è stato definito un Canvas per la visualizzazione grafica degli oggetti, corredato di un Menu laterale, disposto strategicamente al fine di semplificarne l'accesso agli strumenti ed opzioni.

Sulla base degli argomenti teorici estratti dal programma di insegnamento di un istituto tecnico superiore ad indirizzo informatico, sono stati definiti nel dettaglio 6 livelli del gioco articolando per ciascuno di essi gli step che lo compongono, le modalità di interazione con l'utente e gli argomenti teorici trattati. Al fine di coprire in modo esaustivo e non ridondante gli argomenti del 3 anno, si è deciso di sviluppare solo i primi 3 livelli. Successivamente, sono presentati i prototipi a bassa e media fedeltà dei livelli progettati e sono definite le modalità di interazione dei componenti l'utente e le principali funzionalità del gioco, come l'interpretazione del Codice e del Canvas e l'evidenziazione del codice associato agli oggetti selezionati.

Per lo sviluppo del gioco sono state utilizzate diverse tecnologie, basate su React e JavaScript per il front-end e su Spring Boot e Kotlin per il back-end rispettivamente. Per consentire una maggiore portabilità e manutenibilità del codice, inoltre, sono stati realizzati 4 diversi container mediante l'utilizzo di Docker. Sono dunque riportati degli estratti del codice al fine di fornire degli esempi concreti delle implementazioni realizzate.

A seguire, è stato condotto un test di usabilità del gioco coinvolgendo un gruppo di 5 studenti provenienti da diversi istituti scolastici. Il test è stato effettuato online e ha coinvolto l'esecuzione da parte dei partecipanti di 7 task distinti, ciascuno correlato alle rispettive metriche di valutazione. A test concluso, è stato somministrato un questionario post-test SUS ed è stata condotta una fase di debriefing con l'obiettivo di raccogliere il massimo numero possibile di dati.

I risultati raccolti attraverso i test di usabilità hanno soddisfatto le aspettative, dimostrando il successo della piattaforma e del suo approccio innovativo e coinvolgente, comprovato sia dai dati oggettivi raccolti, sia dalle osservazioni dei partecipanti.

"Finalmente ho una interfaccia grafica dove posso vedere cosa sto facendo" afferma il primo utente nella fase di debriefing: questa forte affermazione conferma l'obiettivo della tesi, ovvero che il fornire una rappresentazione grafica di concetti così teorici e difficili da visualizzare possa favorire in modo tangibile l'apprendimento degli studenti.

Ancora, *"La cosa buona è che ho capito che cliccando sul Canvas mi compare a sinistra il codice da scrivere"* afferma il secondo partecipante dei test, dimostrando come la funzionalità di evidenziazione del codice dopo il processo di interpretazione del Canvas, fornisca uno strumento valido ai fini dell'apprendimento.

L'utilità intrinseca del gioco percepita dagli studenti è avvalorata ancora di più dal fatto che tutti i partecipanti del test abbiano affermato che intenderebbero utilizzare il gioco frequentemente.

In conclusione, il gioco Canvas Coding si presenta come un'innovativa risorsa educativa che unisce la creatività alla programmazione, offrendo agli studenti uno strumento coinvolgente ed efficace per apprendere i concetti complessi somministrati durante i tradizionali corsi di introduzione alla programmazione.

6.1 Sviluppo futuro

Durante i test di usabilità condotti, sono emersi spunti riguardo lo sviluppo futuro di nuove funzionalità e modifica di quelle esistenti. A seguire, alcuni esempi:

- **Possibilità di cambiare lingua**: richiesto esplicitamente dai primi 2 partecipanti, potrebbe risultare utile inserire la possibilità di cambiare lingua del gioco. In questo modo, gli studenti possono meglio comprendere i dialoghi presentati nell'Action Menu;

- **Inserire altri linguaggi di programmazione**: come suggerito dagli utenti, uno sviluppo futuro potrebbe prevedere l'inserimento di un selettore che consenta di cambiare il linguaggio di programmazione utilizzato. Questa funzionalità però, trova limitazione nella tipologia di linguaggio utilizzato, in quanto gli argomenti presentati nel gioco si prestano bene a rappresentare i concetti legati ai linguaggi orientati ad oggetti, ma potrebbero risultare meno efficaci in contesti differenti;

- **Fornire degli esempi di codice ed ulteriori suggerimenti**: sebbene i suggerimenti siano presenti ma nessun tester sia stato in grado di accedervi, sicuramente l'inserimento di snippets di codice già scritto a titolo di esempio potrebbe risultare utile agli utenti. *"I suggerimenti per me sono fondamentali"* afferma spesso il secondo partecipante: questo conferma che la presenza dei suggerimenti corrisponde ad un grande aiuto per gli utenti, per questo sarebbe necessario ridefinire il posizionamento o l'intera modalità di visualizzazione attualmente adottata;

- **Aggiungere scorciatoie da tastiera**: domanda sollevata dal quarto partecipante, corrisponde ad un ottimo spunto per una futura implementazione,

consente di ridurre i tempi di esecuzione offrendo agli utenti l'opportunità di "familiarizzare" maggiormente con la piattaforma;

- **Fornire degli esercizi dedicati per le strutture di controllo**: nonostante questi siano presenti nel livello 3, è stato evidenziato come l'attuale implementazione non mostri in modo marcato le loro potenzialità. Sarebbe necessario ridefinire il loro utilizzo, magari mediante l'inserimento di un livello aggiuntivo dedicato a questa tematica cosi importante per i programmatori novizi.

Bibliografia

[1] Dianna Xu, Aaron Cadle, Darby Thompson, Ursula Wolz, Ira Greenberg e Deepak Kumar. «Creative Computation in High School». In: *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*. SIGCSE '16. Memphis, Tennessee, USA: Association for Computing Machinery, 2016, pp. 273–278. ISBN: 9781450336857. DOI: 10.1145/2839509.2844611. URL: https://doi.org/10.1145/2839509.2844611 (cit. alle pp. 2, 7, 13, 16–18, 23, 24, 26, 29).

[2] Sadia Sharmin. «Creativity in CS1: A Literature Review». In: *ACM Trans. Comput. Educ.* 22.2 (nov. 2021). DOI: 10.1145/3459995. URL: https://doi.org/10.1145/3459995 (cit. alle pp. 2, 3, 9, 11, 13, 14, 23–25, 27).

[3] Wouter Groeneveld, Brett A. Becker e Joost Vennekens. «How Creatively Are We Teaching and Assessing Creativity in Computing Education: A Systematic Literature Review». In: *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education - Volume 1*. SIGCSE 2022. Providence, RI, USA: Association for Computing Machinery, 2022, pp. 934–940. ISBN: 9781450390705. DOI: 10.1145/3478431.3499360. URL: https://doi.org/10.1145/3478431.3499360 (cit. alle pp. 2, 3, 9, 12–14, 16, 27, 29).

[4] Ira Greenberg, Deepak Kumar e Dianna Xu. «Creative Coding and Visual Portfolios for CS1». In: *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*. SIGCSE '12. Raleigh, North Carolina, USA: Association for Computing Machinery, 2012, pp. 247–252. ISBN: 9781450310987. DOI: 10.1145/2157136.2157214. URL: https://doi.org/10.1145/2157136.2157214 (cit. alle pp. 2, 3, 6–8, 13, 15, 17, 18, 23, 24, 27, 29).

[5] Catherine Lang, Judy McKay e Sue Lewis. «Seven Factors That Influence ICT Student Achievement». In: *Proceedings of the 12th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*. ITiCSE '07. Dundee, Scotland: Association for Computing Machinery, 2007, pp. 221–225. ISBN: 9781595936103. DOI: 10.1145/1268784.1268849. URL: https://doi.org/10.1145/1268784.1268849 (cit. a p. 2).

[6] Neil Anderson, Colin Lankshear, Carolyn Timms e Lyn Courtney. «'Because it's boring, irrelevant and I don't like computers': Why high school girls avoid professionally-oriented ICT subjects». In: *Computers & Education* 50.4 (2008), pp. 1304–1318. ISSN: 0360-1315. DOI: https://doi.org/10.1016/j.compedu.2006.12.003. URL: https://www.sciencedirect.com/science/article/pii/S0360131506001953 (cit. a p. 2).

[7] Dianna Xu, Ursula Wolz, Deepak Kumar e Ira Greenburg. «Updating Introductory Computer Science with Creative Computation». In: *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*. SIGCSE '18. Baltimore, Maryland, USA: Association for Computing Machinery, 2018, pp. 167–172. ISBN: 9781450351034. DOI: 10.1145/3159450.3159539. URL: https://doi.org/10.1145/3159450.3159539 (cit. alle pp. 2, 6, 18, 23, 29).

[8] Aisling Kelliher. «Technology and the Arts: Educational Encounters of the Third Kind». In: *IEEE MultiMedia* 23.3 (2016), pp. 8–11. DOI: 10.1109/MMUL.2016.41 (cit. alle pp. 3, 12, 13, 22, 25, 27).

[9] Michael Kölling e Poul Henriksen. «Game Programming in Introductory Courses with Direct State Manipulation». In: *SIGCSE Bull.* 37.3 (giu. 2005), pp. 59–63. ISSN: 0097-8418. DOI: 10.1145/1151954.1067465. URL: https://doi.org/10.1145/1151954.1067465 (cit. alle pp. 3, 31).

[10] Juha Sorva, Ville Karavirta e Lauri Malmi. «A Review of Generic Program Visualization Systems for Introductory Programming Education». In: *ACM Trans. Comput. Educ.* 13.4 (nov. 2013). DOI: 10.1145/2490822. URL: https://doi.org/10.1145/2490822 (cit. a p. 3).

[11] Markeya S. Peteranetz, Abraham E. Flanigan, Duane F. Shell e Leen-Kiat Soh. «Helping Engineering Students Learn in Introductory Computer Science (CS1) Using Computational Creativity Exercises (CCEs)». In: *IEEE Transactions on Education* 61.3 (2018), pp. 195–203. DOI: 10.1109/TE.2018.2804350 (cit. alle pp. 3, 10, 11, 13, 16, 27).

[12] Andrew M Mcnutt, Anton Outkine e Ravi Chugh. «A Study of Editor Features in a Creative Coding Classroom». In: *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. CHI '23. Hamburg, Germany: Association for Computing Machinery, 2023. ISBN: 9781450394215. DOI: 10.1145/3544548.3580683. URL: https://doi.org/10.1145/3544548.3580683 (cit. alle pp. 3, 16, 23, 25, 29).

[13] Rodrigo Pessoa Medeiros, Geber Lisboa Ramalho e Taciana Pontual Falcão. «A Systematic Literature Review on Teaching and Learning Introductory Programming in Higher Education». In: *IEEE Transactions on Education* 62.2 (2019), pp. 77–90. DOI: 10.1109/TE.2018.2864133 (cit. alle pp. 3, 13, 14, 24, 27, 29).

[14] "Tammy VanDeGrift". «"Encouraging Creativity In Introductory Computer Science Programming Assignments"». In: *2007 Annual Conference & Exposition*. 10.18260/1-2–1604. https://peer.asee.org/1604. Honolulu, Hawaii: "ASEE Conferences", giu. 2007 (cit. alle pp. 3, 14, 24, 27).

[15] John Maeda e Paola Antonelli. *Design by Numbers*. Cambridge, MA, USA: MIT Press, 1999. ISBN: 0262133547 (cit. a p. 7).

[16] J. Maeda, R. Burns, Thames, Hudson e Massachusetts Institute of Technology. Media Laboratory. *Creative Code*. Thames & Hudson, 2004. ISBN: 9780500285176. URL: https://books.google.it/books?id=VeO6GwAACAAJ (cit. a p. 8).

[17] Gayithri Jayathirtha e Yasmin B. Kafai. «Interactive Stitch Sampler: A Synthesis of a Decade of Research on Using Electronic Textiles to Answer the Who, Where, How, and What for K–12 Computer Science Education». In: *ACM Trans. Comput. Educ.* 20.4 (ott. 2020). DOI: 10.1145/3418299. URL: https://doi.org/10.1145/3418299 (cit. a p. 8).

[18] Ralf Romeike. «Applying Creativity in CS High School Education: Criteria, Teaching Example and Evaluation». In: *Proceedings of the Seventh Baltic Sea Conference on Computing Education Research - Volume 88*. Koli Calling '07. Koli National Park, Finland: Australian Computer Society, Inc., 2007, pp. 87–96. ISBN: 9781920682699 (cit. alle pp. 8, 9, 13, 14, 27).

[19] Sorin Lerner. «Projection Boxes: On-the-Fly Reconfigurable Visualization for Live Programming». In: *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. CHI '20. Honolulu, HI, USA: Association for Computing Machinery, 2020, pp. 1–7. ISBN: 9781450367080. DOI: 10.1145/3313831.3376494. URL: https://doi.org/10.1145/3313831.3376494 (cit. a p. 8).

[20] D. Egbert, G. Bebis, M. McIntosh, N. LaTouttette e A. Mitra. «Computer vision research as a teaching tool in CS1». In: *32nd Annual Frontiers in Education*. Vol. 1. 2002, T4G–T4G. DOI: 10.1109/FIE.2002.1158014 (cit. alle pp. 8, 13).

[21] Alessio Bellino, Valeria Herskovic, Michael Hund e Jorge Munoz-Gama. «A Real-World Approach to Motivate Students on the First Class of a Computer Science Course». In: *ACM Trans. Comput. Educ.* 21.3 (mag. 2021). DOI: 10.1145/3445982. URL: https://doi.org/10.1145/3445982 (cit. alle pp. 8, 13, 14, 27, 29–31).

[22] Jingyi Li, Sonia Hashim e Jennifer Jacobs. «What We Can Learn From Visual Artists About Software Development». In: *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. CHI '21. Yokohama, Japan: Association for Computing Machinery, 2021. ISBN: 9781450380966. DOI: 10.1145/3411764.3445682. URL: https://doi.org/10.1145/3411764.3445682 (cit. alle pp. 8, 13, 29).

[23] Arnon Hershkovitz, Raquel Sitman, Pablo Garaizar Rotem Israel-Fishelson Andoni Eguíluz e Mariluz Guenaga. «Creativity in the acquisition of computational thinking». In: *Interactive Learning Environments* 27 (2019). DOI: 10.1080/10494820.2019.1610451. eprint: https://doi.org/10.1080/10494820.2019.1610451. URL: https://doi.org/10.1080/10494820.2019.1610451 (cit. alle pp. 9, 10, 31).

[24] Andrea Salgian, Teresa M. Nakra, Christopher Ault e Yunfeng Wang. «Teaching Creativity in Computer Science». In: *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*. SIGCSE '13. Denver, Colorado, USA: Association for Computing Machinery, 2013, pp. 123–128. ISBN: 9781450318686. DOI: 10.1145/2445196.2445238. URL: https://doi.org/10.1145/2445196.2445238 (cit. a p. 9).

[25] David Spendlove. «Creativity in education : a review». In: 10 (mag. 2008) (cit. alle pp. 10, 11).

[26] Steven M. Schmidt Robert Epstein e Regina Warfel. «Measuring and Training Creativity Competencies: Validation of a New Test». In: *Creativity Research Journal* 20.1 (2008), pp. 7–12. DOI: 10.1080/10400410701839876. eprint: https://doi.org/10.1080/10400410701839876. URL: https://doi.org/10.1080/10400410701839876 (cit. a p. 10).

[27] Qiang Li, Ze-xue Liu, Peng Wang, Jing-jing Wang e Tian Luo. «The influence of art programming courses on design thinking and computational thinking in college art and design students». In: *Education and Information Technologies* 28.9 (set. 2023), pp. 10885–10902. ISSN: 1573-7608. DOI: 10.1007/s10639-023-11618-7. URL: https://doi.org/10.1007/s10639-023-11618-7 (cit. alle pp. 10, 18).

[28] R. Keith Sawyer. «Educating for innovation». In: *Thinking Skills and Creativity* 1.1 (2006), pp. 41–48. ISSN: 1871-1871. DOI: https://doi.org/10.1016/j.tsc.2005.08.001. URL: https://www.sciencedirect.com/science/article/pii/S1871187105000052 (cit. a p. 10).

[29] Mark Anderson e Collette Gavan. «Engaging Undergraduate Programming Students: Experiences Using Lego Mindstorms NXT». In: *Proceedings of the 13th Annual Conference on Information Technology Education*. SIGITE '12. Calgary, Alberta, Canada: Association for Computing Machinery, 2012,

pp. 139–144. ISBN: 9781450314640. DOI: 10.1145/2380552.2380595. URL: https://doi.org/10.1145/2380552.2380595 (cit. a p. 14).

[30] Ruanqianqian (Lisa) Huang, Kasra Ferdowsi, Ana Selvaraj, Adalbert Gerald Soosai Raj e Sorin Lerner. «Investigating the Impact of Using a Live Programming Environment in a CS1 Course». In: *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education - Volume 1*. SIGCSE 2022. Providence, RI, USA: Association for Computing Machinery, 2022, pp. 495–501. ISBN: 9781450390705. DOI: 10.1145/3478431.3499305. URL: https://doi.org/10.1145/3478431.3499305 (cit. alle pp. 14, 27).

[31] Michail N. Giannakos, Letizia Jaccheri e Roberta Proto. «Teaching Computer Science to Young Children through Creativity: Lessons Learned from the Case of Norway». In: *Proceedings of the 3rd Computer Science Education Research Conference on Computer Science Education Research*. CSERC '13. Arnhem, Netherlands: Open Universiteit, Heerlen, 2013, pp. 103–111 (cit. alle pp. 15, 26, 27).

[32] Leo Porter, Mark Guzdial, Charlie McDowell e Beth Simon. «Success in Introductory Programming: What Works?» In: *Commun. ACM* 56.8 (ago. 2013), pp. 34–36. ISSN: 0001-0782. DOI: 10.1145/2492007.2492020. URL: https://doi.org/10.1145/2492007.2492020 (cit. alle pp. 16, 17, 23, 27).

[33] *L. McCarthy and M. Turner. [n.d.]. p5.js.* https://p5js.org/ (cit. a p. 18).

[34] *ofBook, a collaboratively written book about openFrame- works.* http://openframeworks.cc/ofBook/chapters/foreword.html (cit. a p. 20).

[35] Zoe J. Wood, Paul Muhl e Katelyn Hicks. «Computational Art: Introducing High School Students to Computing via Art». In: *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*. SIGCSE '16. Memphis, Tennessee, USA: Association for Computing Machinery, 2016, pp. 261–266. ISBN: 9781450336857. DOI: 10.1145/2839509.2844614. URL: https://doi.org/10.1145/2839509.2844614 (cit. alle pp. 24, 29).

[36] Christopher D. Hundhausen, Sean F. Farley e Jonathan L. Brown. «Can Direct Manipulation Lower the Barriers to Computer Programming and Promote Transfer of Training? An Experimental Study». In: *ACM Trans. Comput.-Hum. Interact.* 16.3 (set. 2009). ISSN: 1073-0516. DOI: 10.1145/1592440.1592442. URL: https://doi.org/10.1145/1592440.1592442 (cit. alle pp. 27, 29).

[37] James D. Hollan Edwin L. Hutchins e Donald A. Norman. «Direct Manipulation Interfaces». In: *Human–Computer Interaction* 1.4 (1985), pp. 311–338. DOI: 10.1207/s15327051hci0104_2. eprint: https://doi.org/10.1207/s15327051hci0104_2. URL: https://doi.org/10.1207/s15327051hci0104_2 (cit. a p. 31).