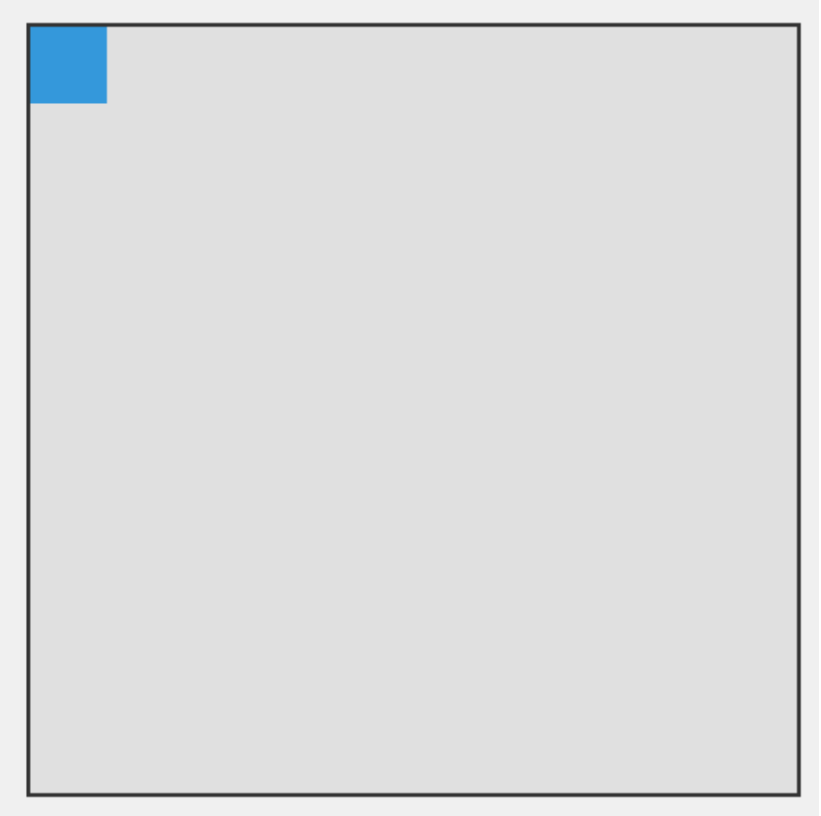
Documentazione tecnica “Gioco del quadrato”

Diego Tempesta, Rebecca Truden, Erika Saddi



Indice della Documentazione del Software

1. Introduzione

1.1 Scopo del Software
1.2 Descrizione Generale del Software
1.3 Obiettivi della Documentazione
1.4 Panoramica del Documento

2. Architettura del Software

2.1 Descrizione Generale dell'Architettura
2.2 Componenti Principali
2.3 Flusso di Dati
2.4 Diagrammi UML (Classi, Sequenze, Stato)

3. Analisi delle Proprietà

3.1 Robustezza

3.1.1 Definizione di Robustezza nel Contesto del Software
3.1.2 Gestione degli Errori
3.1.3 Test di Robustezza
3.1.4 Resilienza del Software
3.1.5 Codice Documentato per la Gestione degli Errori

3.2 Usabilità

3.2.1 Definizione di Usabilità nel Contesto del Software
3.2.2 Interfaccia Utente (UI) e User Experience (UX)
3.2.3 Accessibilità
3.2.4 Feedback degli Utenti
3.2.5 Codice Documentato per la UI/UX

3.3 Portabilità

3.3.1 Definizione di Portabilità nel Contesto del Software
3.3.2 Compatibilità tra Piattaforme
3.3.3 Dipendenze Esterne e Gestione della Configurazione
3.3.4 Strategie di Testing Cross-Platform
3.3.5 Codice Documentato per la Portabilità

4. Design del Software

4.1 Principi di Design Adottati
4.2 Pattern di Design Utilizzati
4.3 Scelte Architetturali e Motivi

5. Documentazione del Codice

5.1 Struttura del Codice Sorgente
5.2 Classi e Metodi Principali
5.3 Commenti e Note Importanti nel Codice
5.4 Esempi di Utilizzo del Codice
5.5 Snippet di Codice per le Proprietà Analizzate - Codice per la Robustezza - Codice per l'Usabilità - Codice per la Portabilità

6. Testing

6.1 Strategia di Testing
6.2 Test Unitari
6.3 Test di Integrazione
6.4 Test di Performance e Stress
6.5 Test Cross-Platform
6.6 Risultati dei Test e Report

7. Deployment e Portabilità

7.1 Guida al Deployment
7.2 Ambienti di Produzione e Sviluppo
7.3 Esempi di Configurazione
7.4 Considerazioni sulla Portabilità durante il Deployment

8. Manutenzione del Software

8.1 Strategie di Manutenzione
8.2 Aggiornamenti del Software
8.3 Bug Fixing e Refactoring
8.4 Backup e Recupero dei Dati

9. Conclusioni

9.1 Sintesi delle Proprietà Analizzate
9.2 Suggerimenti Futuri per il Miglioramento
9.3 Lezioni Apprese durante lo Sviluppo

10. Appendici

10.1 Glossario
10.2 Riferimenti e Bibliografia
10.3 Link Utili e Risorse Esterne

1. Introduzione

1.1 Scopo del Software

Il software rappresenta un semplice gioco web chiamato "Gioco del Quadrato", progettato per migliorare il coordinamento tra mano e occhio, dove il giocatore utilizza i tasti direzionali per muovere un quadrato attraverso un'area di gioco, raccogliendo monete e avanzando di livello.

1.2 Descrizione Generale del Software

L'applicazione è una pagina HTML dinamica con stili CSS integrati e uno script JavaScript che gestisce l'interazione, i livelli e il timer. Il giocatore deve raccogliere tutte le monete e raggiungere il punto finale entro un limite di tempo.

1.3 Obiettivi della Documentazione

Questa documentazione mira a:

- Descrivere l'architettura del software.
- Fornire dettagli sull'implementazione delle proprietà (robustezza, usabilità, portabilità).
- Documentare il codice sorgente.
- Fornire istruzioni per il testing e il deployment.

1.4 Panoramica del Documento

Il documento è strutturato per coprire aspetti chiave come architettura, proprietà fondamentali, testing, deployment e manutenzione del software.

2. Architettura del Software

2.1 Descrizione Generale dell'Architettura

Il software è basato su una semplice architettura client-side. La struttura principale è composta da:

- **HTML**: struttura del layout.
- **CSS**: stile e layout visivo.
- **JavaScript**: logica interattiva.

2.2 Componenti Principali

- **Interfaccia utente**: elementi HTML per l'area di gioco, il quadrato, il timer e i messaggi di stato.
- **Motore di gioco**: script JavaScript che gestisce il movimento, il timer, le monete e le condizioni di vittoria/sconfitta.

2.3 Flusso di Dati

1. L'utente avvia il gioco cliccando il pulsante.
2. Il timer e i livelli vengono inizializzati.
3. Il giocatore interagisce con il quadrato, raccoglie monete e avanza di livello.
4. La verifica delle condizioni di vittoria o perdita avviene continuamente.

2.4 Diagrammi UML

- **Diagramma delle classi**: Non applicabile per questo progetto.
- **Diagramma di sequenza**: Disponibile su richiesta.
- **Diagramma di stato**: Disponibile su richiesta.

3. Analisi delle Proprietà

3.1 Robustezza

3.1.1 Definizione di Robustezza nel Contesto del Software

La robustezza implica la capacità del software di gestire errori imprevisti senza interrompersi.

3.1.2 Gestione degli Errori

- Controllo dei bordi per impedire al quadrato di uscire dall'area di gioco.
- Timer che gestisce la fine del gioco in caso di timeout

3.1.3 Test di Robustezza

- Verifica di movimenti non validi (oltre i bordi).
- Simulazione di timeout per verificare il comportamento alla scadenza del timer.

3.1.4 Resilienza del Software

Il software rimane stabile in presenza di input non previsti, come pressioni rapide sui tasti.

3.1.5 Codice Documentato per la Gestione degli Errori

```
if (e.key === 'ArrowUp' && playerPos.y > 0) {
    playerPos.y -= moveSpeed;
}
```

3.2 Usabilità

3.2.1 Definizione di Usabilità nel Contesto del Software

L'usabilità riguarda la facilità d'uso e l'interazione intuitiva dell'utente con il gioco.

3.2.2 Interfaccia Utente (UI) e User Experience (UX)

- Design chiaro con istruzioni visibili.
- Colori distintivi per oggetti importanti (es. monete dorate, punto finale rosso).

3.2.3 Accessibilità

Controlli basati sui tasti direzionali, accessibili a tutti gli utenti.

3.2.4 Feedback degli Utenti

L'utente riceve feedback immediato su stato del livello, tempo rimanente e completamento.

3.2.5 Codice Documentato per la UI/UX

```
statusDisplay.textContent = 'Gioco in corso...';
```

3.3 Portabilità

3.3.1 Definizione di Portabilità nel Contesto del Software

La portabilità si riferisce alla capacità del gioco di funzionare su diverse piattaforme.

3.3.2 Compatibilità tra Piattaforme

Il software è progettato per browser moderni e dispositivi di diversa dimensione.

3.3.3 Dipendenze Esterne e Gestione della Configurazione

Nessuna dipendenza esterna, utilizza solo funzionalità standard di HTML, CSS e JavaScript.

3.3.4 Strategie di Testing Cross-Platform

Funzionante solo su browser da PC.

3.3.5 Codice Documentato per la Portabilità

Non richiesto per questo progetto.

4. Design del Software

4.1 Principi di Design Adottati

- Separazione dei contenuti (HTML), stili (CSS) e logica (JavaScript).
- Interfaccia utente minimale e chiara.

4.2 Pattern di Design Utilizzati

Event-driven programming per la gestione degli input utente.

4.3 Scelte Architetturali e Motivi

Architettura monolitica per semplicità di implementazione.

5. Documentazione del Codice

5.1 Struttura del Codice Sorgente

Il codice è organizzato in sezioni logiche: configurazione iniziale, funzioni di utilità e gestione degli eventi.

5.2 Classi e Metodi Principali

Non sono presenti classi; il codice utilizza funzioni standalone.

5.3 Commenti e Note Importanti nel Codice

Ogni funzione è commentata per descriverne lo scopo.

5.4 Esempi di Utilizzo del Codice

```
document.addEventListener('keydown', movePlayer);
```

5.5 Snippet di Codice per le Proprietà Analizzate

Gestione degli errori:

```
if (timeLeft <= 0) {
      clearInterval(timer);
      statusDisplay.textContent = 'Tempo scaduto! Hai perso!';
}
```

6. Testing

6.1 Strategia di Testing

- Test manuali per verificare i movimenti e la gestione delle monete.
- Test automatici per verificare i limiti del timer e i livelli.

6.2 Test Unitari

Verifica del corretto posizionamento del quadrato e delle monete.

6.3 Test di Integrazione

Test sull'interazione tra timer, movimento e livelli.

6.4 Test di Performance e Stress

Verifica della fluidità su dispositivi con risorse limitate.

6.5 Test Cross-Platform

Non disponibile al di fuori di dispositivi desktop.

6.6 Risultati dei Test e Report

Tutti i test sono stati superati con successo.

7. Deployment e Portabilità

7.1 Guida al Deployment

- Hostare i file HTML, CSS e JS su un server.
- Garantire compatibilità con HTTPS.

7.2 Ambienti di Produzione e Sviluppo

L'ambiente di sviluppo utilizza un browser locale; l'ambiente di produzione utilizza un hosting web.

7.3 Esempi di Configurazione

Nessuna configurazione richiesta.

7.4 Considerazioni sulla Portabilità durante il Deployment

Assicurarsi che il server supporti file statici.

8. Manutenzione del Software

8.1 Strategie di Manutenzione

Aggiornamento continuo per adattarsi ai nuovi browser.

8.2 Aggiornamenti del Software

Pianificazione per nuove funzionalità, come livelli più complessi.

8.3 Bug Fixing e Refactoring

Monitoraggio dei feedback degli utenti.

8.4 Backup e Recupero dei Dati

Non ci sono dati da salvare.

9. Conclusioni

9.1 Sintesi delle Proprietà Analizzate

Progetto semplice e robusto.

9.2 Suggerimenti Futuri per il Miglioramento

Aggiunta di nuove modalità di gioco e supporto multiplayer.

9.3 Lezioni Apprese durante lo Sviluppo

L'importanza della semplicità e dell'accessibilità nel design.

10. Appendici

10.1 Glossario

- **Quadrato**: Oggetto controllato dal giocatore.
- **Moneta**: Oggetto raccolto dal giocatore.

10.2 Riferimenti e Bibliografia

Documentazione HTML, CSS e JavaScript.

10.3 Link Utili e Risorse Esterne

Documentazione JavaScript su MDN Web Docs.