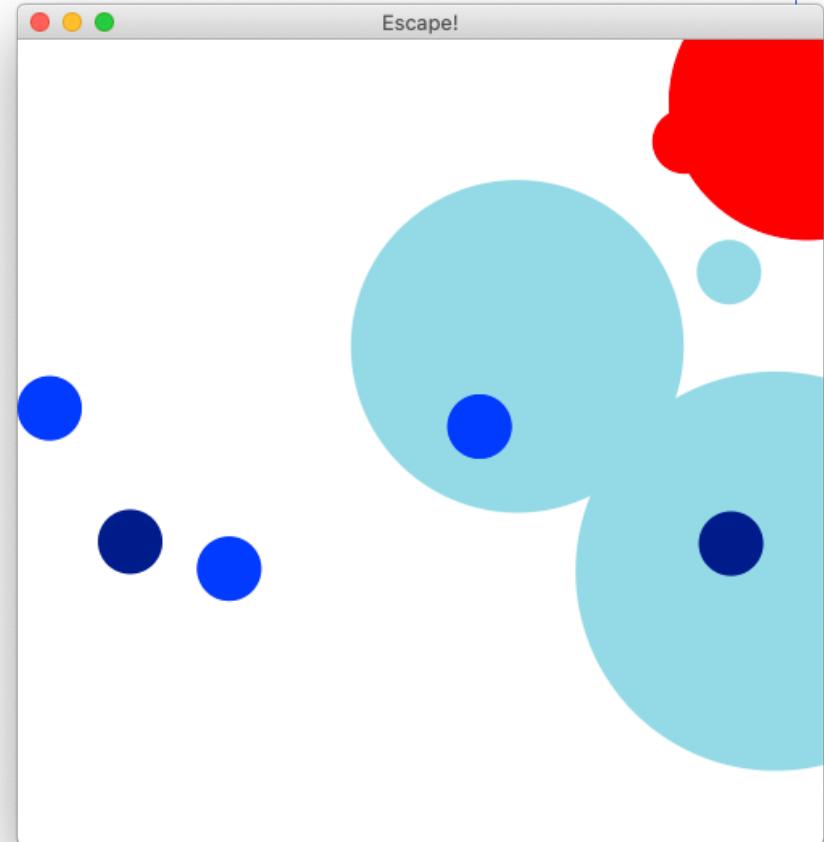


Laboratorio 5

Gian Pietro Picco

Escape!

- ✖ Leggete bene il testo dell'esame
- ✖ Disegnate il class diagram UML
 - identificate la gerarchia di ereditarietà
 - distinguete i metodi specifici di ogni classe e quelli ereditati e/o ridefiniti
- ✖ Pensate alle strutture dati
 - dove immagazzinare le palline attualmente presenti nel gioco?
- ✖ Pensate agli event handler
 - quali eventi vi serve gestire?
 - quali sono le reazioni corrispondenti?



Implementazione

- ✖ Abbiamo implementato alcune funzionalità chiave nel Laboratorio 4 ...
- ✖ ... ma attenzione:
 - le abbiamo implementate direttamente nell'applicazione principale
 - qui dovremo invece inserire/adattare tale codice in modo che sia all'interno di una classe
 - ◆ pallina utente e/o nemica
- ✖ Decisione chiave: rappresentare una pallina
 - soluzione 1:
una classe che contiene fra gli attributi un'istanza di **Circle**
 - soluzione 2:
una classe che estende direttamente da **Circle**

Procedete per gradi!

✗ Ad esempio:

- Cominciate implementando la singola pallina giocatore, sulla base del lab 4
 - ◆ movimento tasti + wrap around bordi schermo
- Aggiungete un nemico e il suo comportamento
- Realizzate il rilevamento della collisione fra giocatore e nemico
- Implementate i nemici rimanenti
- Per finire, implementate la seconda finestra e la relativa gestione del punteggio

✗ Usate la console per il debugging!

Altri dettagli implementativi

- ✖ Per generare numeri casuali: potete usare la classe **Random** (in `java.util`)
 - fornisce metodi per generare
 - ◆ un **int** (tra 0 e un massimo desiderato)
 - ◆ un **double** (tra 0 e 1)
- ✖ Non c'è bisogno di layout particolari
 - può essere utile usarne uno molto semplice nella finestra secondaria del punteggio