

Sopravvivere al codice

Principi e buone pratiche per sopravvivere alla complessità dello sviluppo

Moreno Lovato  **RUNEHEADS**



Ministero delle Imprese
e del Made in Italy



SOPRAVVIVERE AL CODICE

Un viaggio tra principi, pattern e buone pratiche per sopravvivere alla complessità dello sviluppo del vostro progetto.



RUNEHEADS

Moreno Lovato

PRESENTAZIONI

1980s

Primo gioco in assembler su Commodore 64



1990s - 2000s

Esperienze in tutti i settori: progettista di reti, sviluppatore, project manager, IT Manager



2009

Ritorno ai videogiochi

2018

Owner dello studio indipendente RuneHeads



MORENO LOVATO
mlovato@runeheads.com

~20 GIOCHI
~30 ANNI DI ESPERIENZA
~15 ANNI NEL GAMING

Specializzazioni

- Architettura Software
- Unity 3D
- Formazione
- Fisica
- Matematica



SINCE 2017

ESPERIENZE PER ADV E GAMIFICATION

Progetti interattivi su misura per campagne pubblicitarie (ADV) e strategie di gamification.

FORMAZIONE

Docenze presso le maggiori accademie italiane di Gaming

CONSULENZA UNITY

Ottimizzazione processi di sviluppo, definizione basi architetture, sviluppo di features specifiche e porting multi-piattaforma per altri studi

WHITE LABEL

Sviluppo di videogiochi per terze parti. Realizziamo progetti su commissione con standard qualitativi elevati.



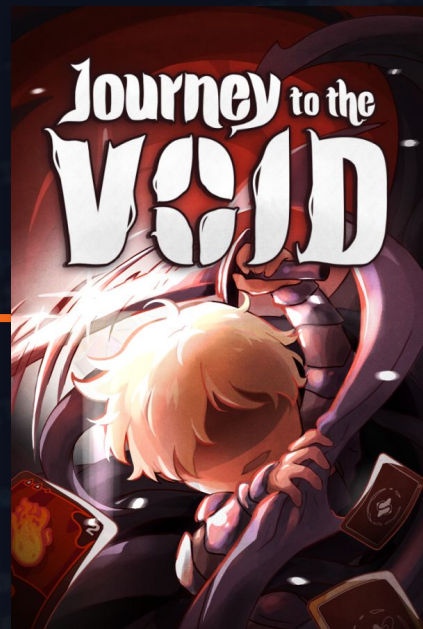
SINCE 2017



2018



2020



2026

ARCHITETTURA SOFTWARE

16 ore in 45 minuti (Sì, respireremo poco)

Un breve spaccato del mio corso di architettura, che normalmente dura 16 ore.

PATTERN DI PROGRAMMAZIONE

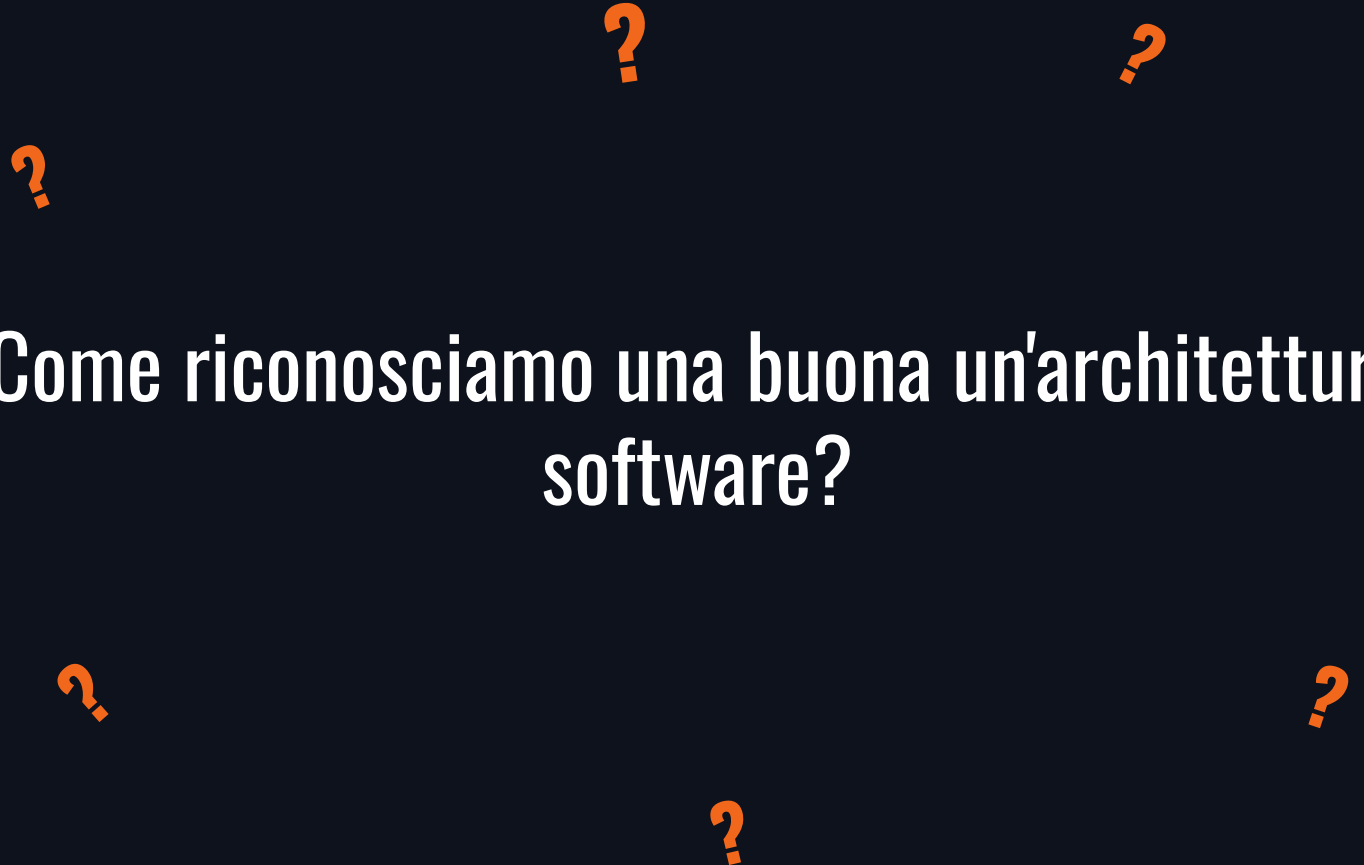
ELENCO E RISORSE PER DIVENTARE PRATICI
DEI PATTERN TIPICI

PRINCIPI ARCHITETTURALI

APPROFONDIMENTO SU BEST PRACTICES E
METODOLOGIE

REDFLAGS

SEGNALI COMUNI DI PROBLEMI
ARCHITETTURALI



Come riconosciamo una buona un'architettura
software?



È A SERVIZIO DELLA CREATIVITÀ



ACCOGLIE IL CAMBIAMENTO!

!?

?!?

?!?

PROGRAMMARE

l'atto di pianificare e organizzare in anticipo una serie di azioni per raggiungere un determinato obiettivo.

?!?

?!?

!?

La falsa illusione del programmatore

Non possiamo prevedere tutto. Dobbiamo essere pronti a cambiare velocemente e spesso!

*L'interfaccia doveva essere minimal...
poi è diventata un cruscotto di Boeing
747.*

*“Il boss finale ora deve volare”
(dopo sei mesi di nemici terrestri)*

*Il game designer ha detto: è solo un
piccolo tweak.*

Il tutorial è cambiato di nuovo.

La falsa illusione del programmatore

Non possiamo prevedere tutto. Dobbiamo essere pronti a cambiare velocemente e spesso!

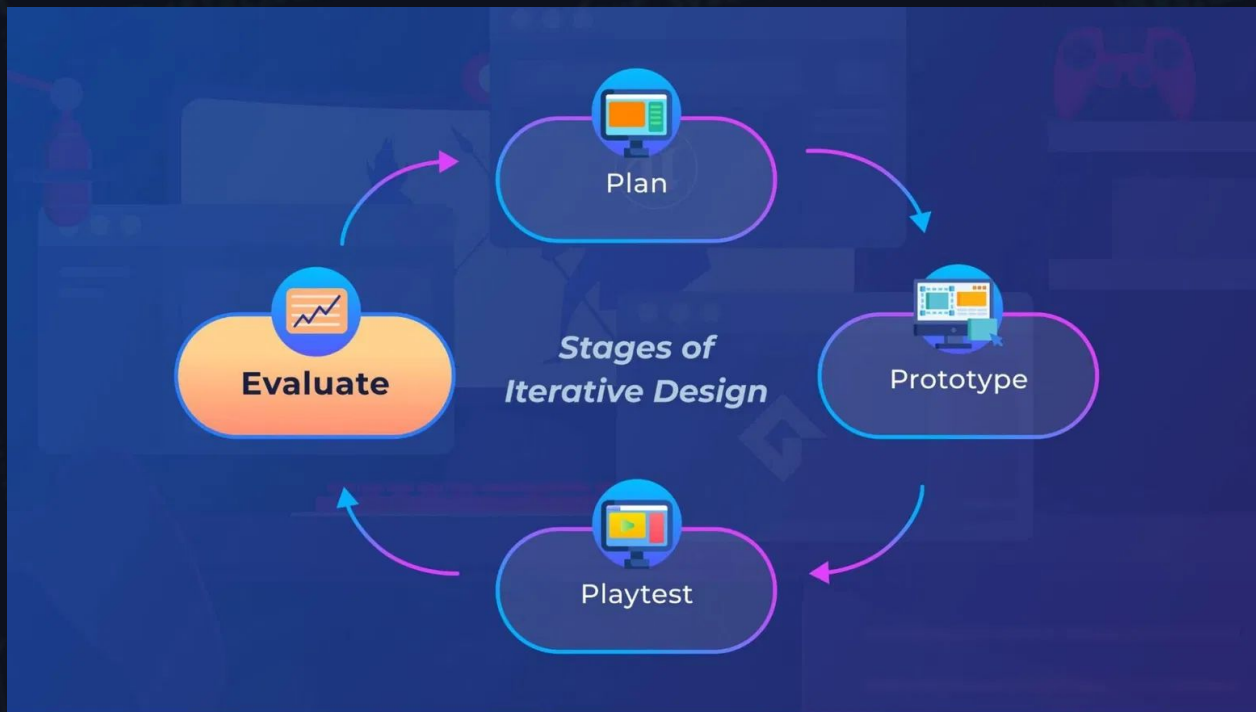
80% DEL CODICE VIENE MODIFICATO
ALMENO UNA VOLTA

LA PARTE CORE DEL GIOCO VERRÀ TOCCATA
ALMENO **20-50** VOLTE



Il cambiamento è necessario

Il cambiamento è intrinseco nella natura dei processi creativi



Una buona architettura

Creatività Libera

Sperimentazione senza limiti tecnici

Velocità

Modifiche in ore, non giorni

Prevedibilità

Nessun effetto collaterale inatteso

Fiducia nel Codice

Zero paura di modificare

Controllo Totale

Complessità sotto controllo

Una buona architettura

Creatività Libera

Sperimentazione senza limiti tecnici

Velocità

Modifiche in ore, non giorni

Prevedibilità

Nessun effetto collaterale inatteso

Fiducia nel Codice

Zero paura di modificare

Controllo Totale

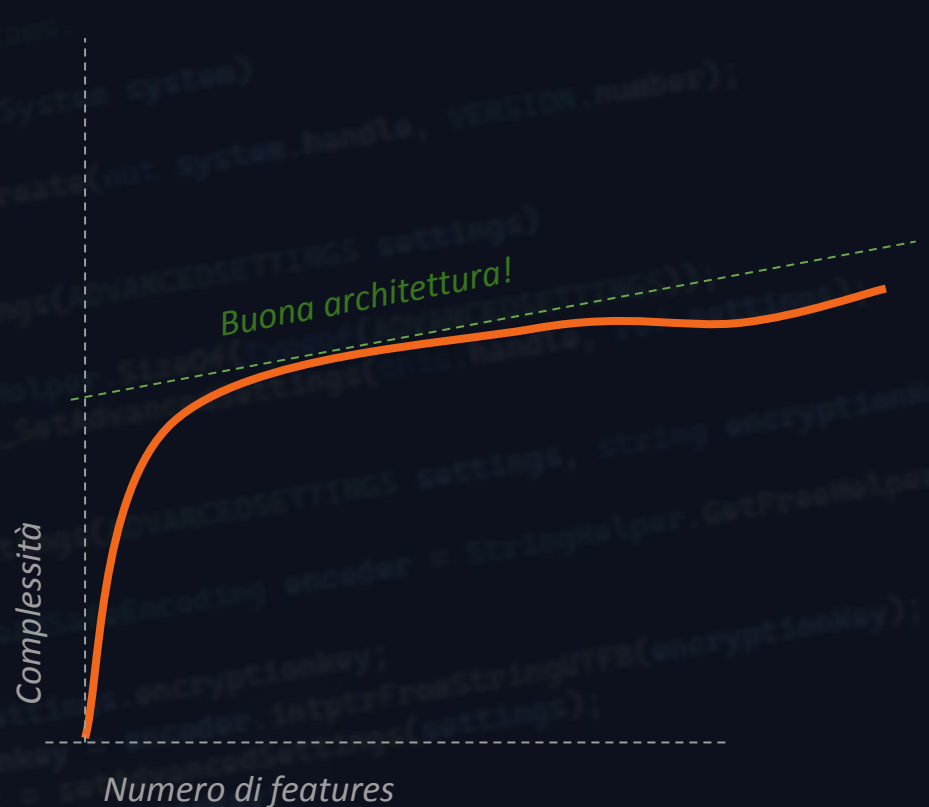
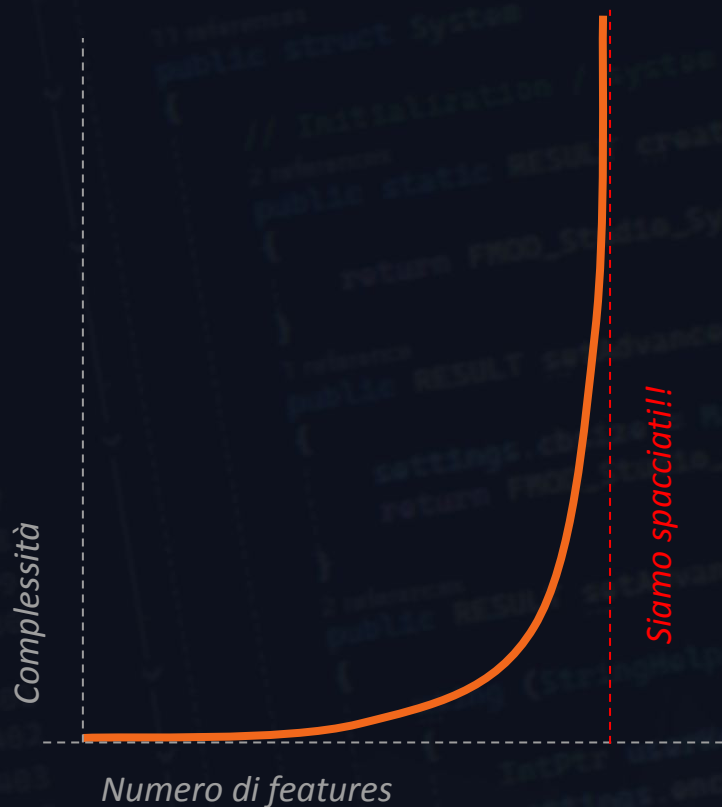
Complessità sotto controllo

- If your code works fine don't touch it
+ my code:



Don't touch it!

La curva della complessità





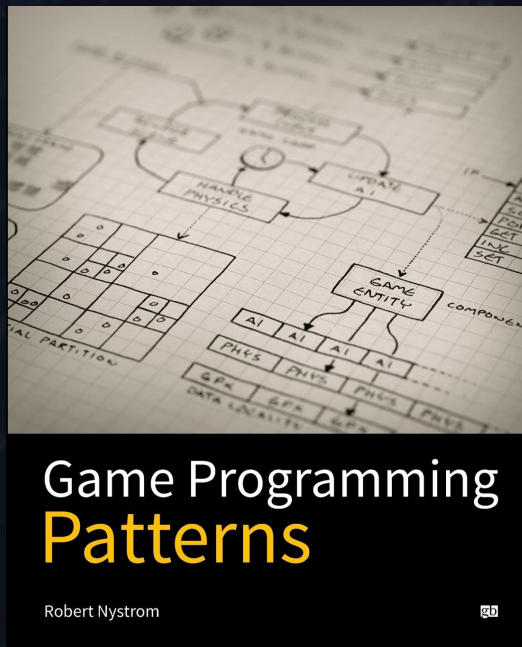
Come realizzo una buona un'architettura software?



PATTERN FONDAMENTALI

- Command
- State
- Flyweight
- Observer
- Prototype
- Singleton
- Service Locator
- TypeObject
- Component
- Event-Queue
- Object-Pool
- MVC
- PSD-MVC (*)

<https://gameprogrammingpatterns.com/contents.html>



PSD-MVC

Persistence

Rappresenta le informazioni minime sullo **STATO** di una entità

Session

È la **COLLEZIONE DI MODEL** che rappresenta il mondo gioco

Database

Rappresenta le informazioni **IMMUTABILI** di un'entità del mondo di gioco

Model

Rappresenta le informazioni **MUTEVOLI** di un'entità nel mondo di gioco

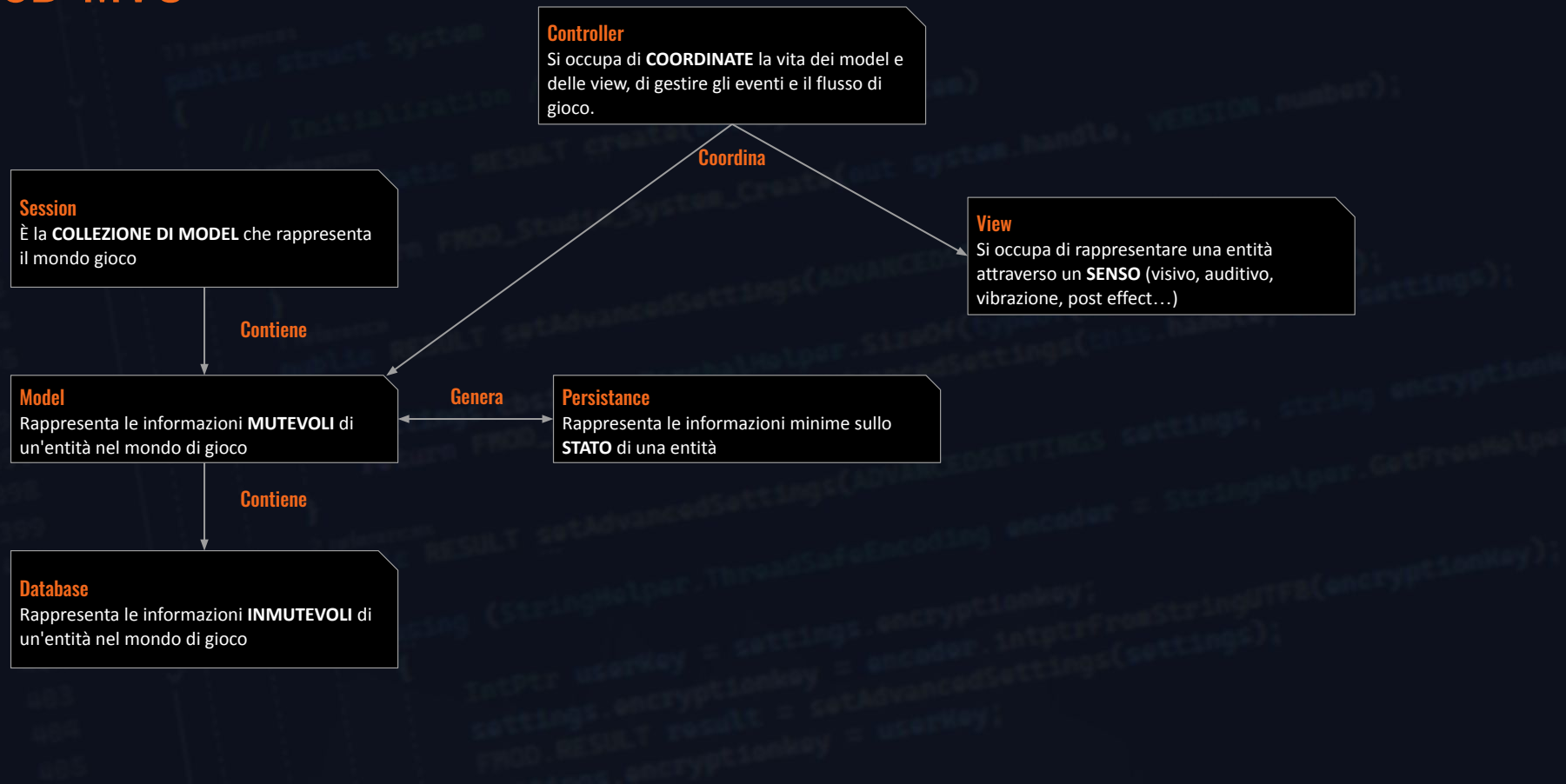
View

Si occupa di rappresentare una entità attraverso un **SENSO** (visivo, auditivo, vibrazione, post effect...)

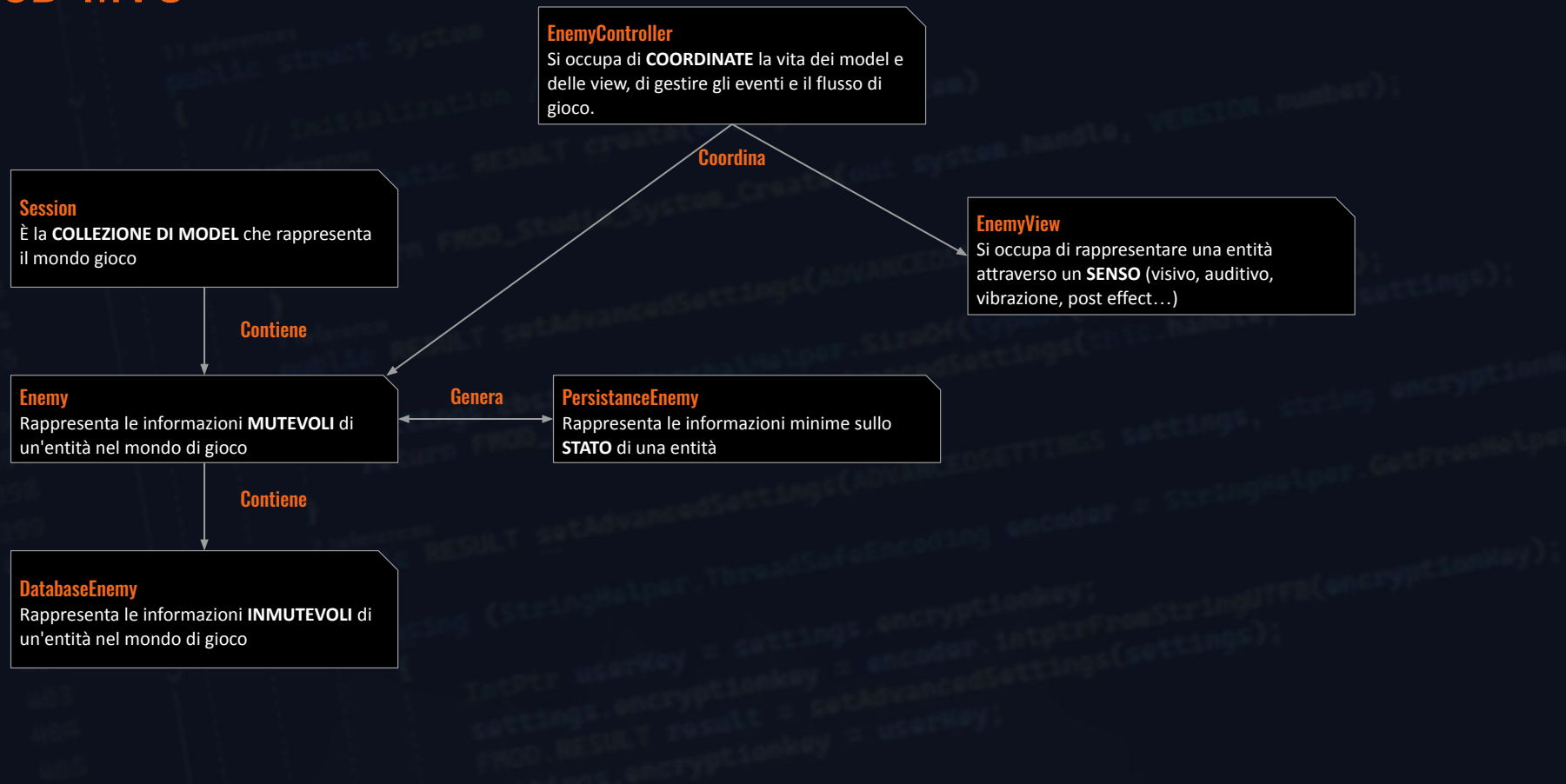
Controller

Si occupa di **COORDINARE** la vita dei model e delle view, di gestire gli eventi e il flusso di gioco.

PSD-MVC

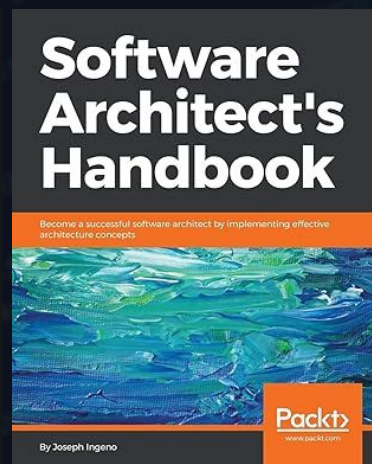
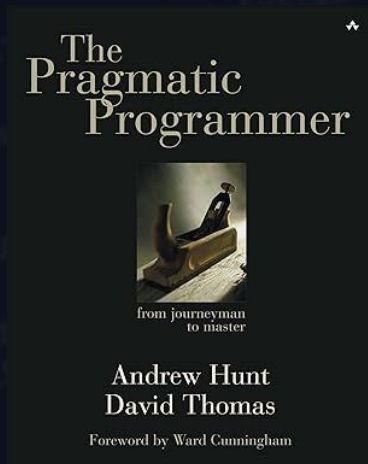


PSD-MVC



PRINCIPI FONDAMENTALI

- PHL (*)
- KISS
- YAGNI
- DRY
- SSOT
- SRP
- DD
- DTA/LoD
- CITI (*)
- ...
- STATELESS/STATEFULL
- WYSIWYG
- SOLID
- ...



PHL

PROGRAM WITH HUMAN LANGUAGE

Librerie & Utilities

Possono parlare in un gergo tecnico

Core Game Code

Parla nel linguaggio del gioco!

SetupAnimator()
CalculatePoints()
RemoveFromArray()

Die()
GetTotalArmon()
Spawn()

(*) Moreno Made

KISS - YAGNI

KEEP IT SHORT AND SIMPLE
YOU AREN'T GONNA NEED IT



CLASSI PICCOLE!

Non superare 100/150 righe di codice

NON CERCARE DI PREVEDERE TUTTO

Perdi tempo e non sarà mai come lo stai prevedendo

80% DEL CODICE VIENE MODIFICATO
ALMENO UNA VOLTA

LA PARTE CORE DEL GIOCO VERRÀ TOCCATA
ALMENO 20-50 VOLTE



DRY - SSOT - SRP

DON'T REPEAT YOURSELF
SINGLE SOURCE OF TRUTH
SINGLE RESPONSIBILITY PRINCIPLE

NON RIPETERE MAI LA STESSA RIGA DI CODICE

Crea possibili BUG, perdi tempo, non saprai più chi ha ragione

UN SOLO RESPONSABILE

Delega la responsabilità di un'azione o controllo ad una sola entità!

`Animator.CanJumpNow()`
`Controller.IsJumping()`

UNA SOLA FONTE DI INFORMAZIONE

Non creare duplicati di una informazione, o informazioni che possono essere incoerenti

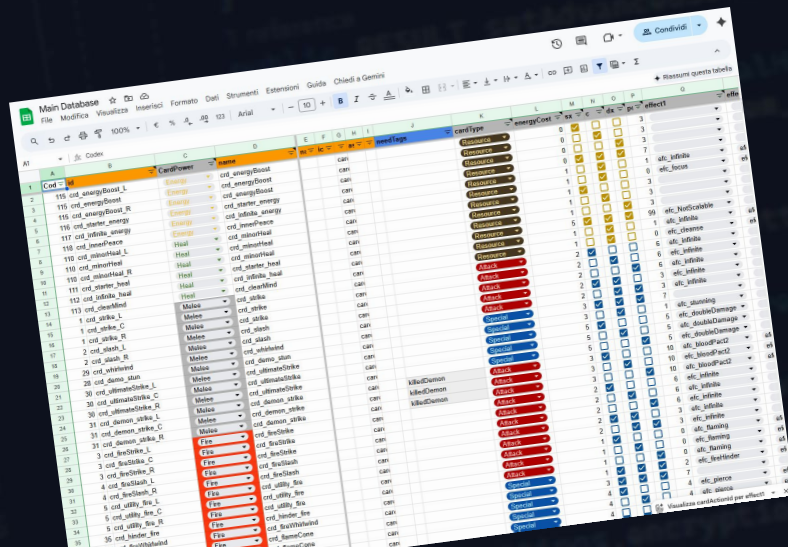
`IsJumping`
`IsRunning`
`IsWalking`

DD

DATA DRIVEN

PROGRAMMA CON I DATI, NON CON IL CODICE

Crea programmi che possano mutare in funzione dei dati. Non includere informazioni statiche nel codice



The screenshot shows a spreadsheet application with a large table of data. The table has columns for 'Code', 'Name', and various attributes. The data is organized into rows, with some rows highlighted in yellow. The spreadsheet is titled 'Main Database' and has a 'Riassunto questo foglio' (Summary of this sheet) button in the top right corner. The table contains a mix of text and numerical data, with some cells containing formulas or references.



EVITA GLI ENUM COME EVITI LA MORTE!

Sono la morte del DD e spesso causa di blocchi architetturali!

“Dai al tuo Designer un foglio di calcolo e lo farai felice!”

DTA/LoD

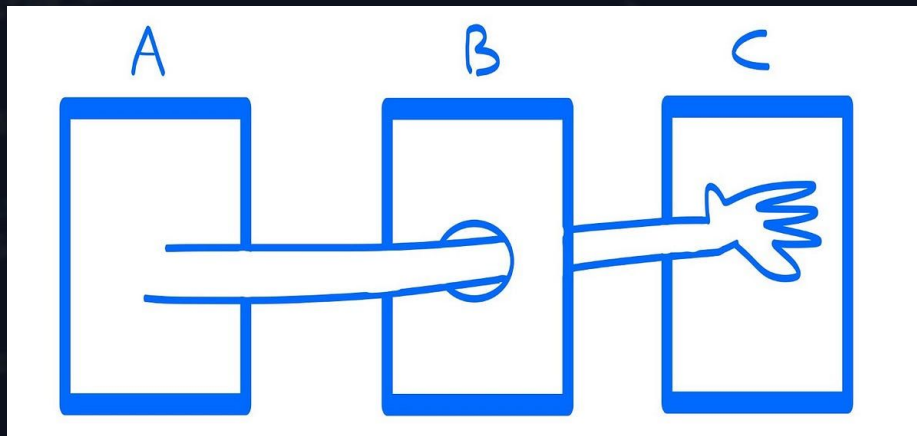
DON'T TELL, ASK
LAW OF DEMETRA

NON ENTRARE IN CASA DI ALTRI!

Parla solo con le classi a te vicine. Non usarle per accedere a loro informazioni o metodi

```
View.TextField.Text = "Ciao"
```

```
View.SetTitle("Ciao")
```



CITI (*)

CAN I TEST IT?

PENSA AL TEST QUANDO PROGETTI

Rendi i tuoi sistemi testabili, svincolati e non interdipendenti



USANDO LA PSD-MVC PUOI SIMULARE UNO STATO DEL GIOCO!

RED FLAGS



REDFLAGS

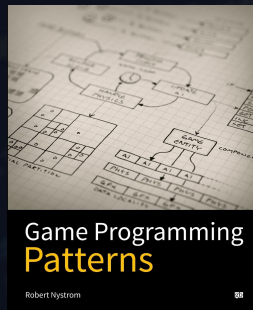
- Bool Usati per stati
- Uso di ENUM
- Singleton
- Variabili con suffissi numerici
- Switch Case
- Classi sopra le 100 righe
- Dot.Dot.Dot
- Incapacità di fare test puntuali
- Non possiamo lavorarci in due
- Circolarità delle dipendenze
- Funzioni identiche in diversi punti
- Nessuna separazione tra logica e dati
- Modifiche multiple per singolo cambiamento logico
- Approccio algoritmico e non di flusso
- Inizializzazioni in Awake/Start
- Funziona, non toccarlo!



TAKEAWAYS

The word "TAKEAWAYS" is centered in a bold, white, sans-serif font. Surrounding it are six orange exclamation marks, each with a small square base, arranged in a circular pattern at the top, bottom, and sides of the frame.

TAKEAWAYS



PHL, KISS, YAGNI

Scrivi classe piccole, scrivi solo quello che serve adesso. Pensa e scrivi con un linguaggio del gioco e con aggettivi e verbi naturale.

CITI

Progetta sistemi testabili. Crea meno dipendenze possibili

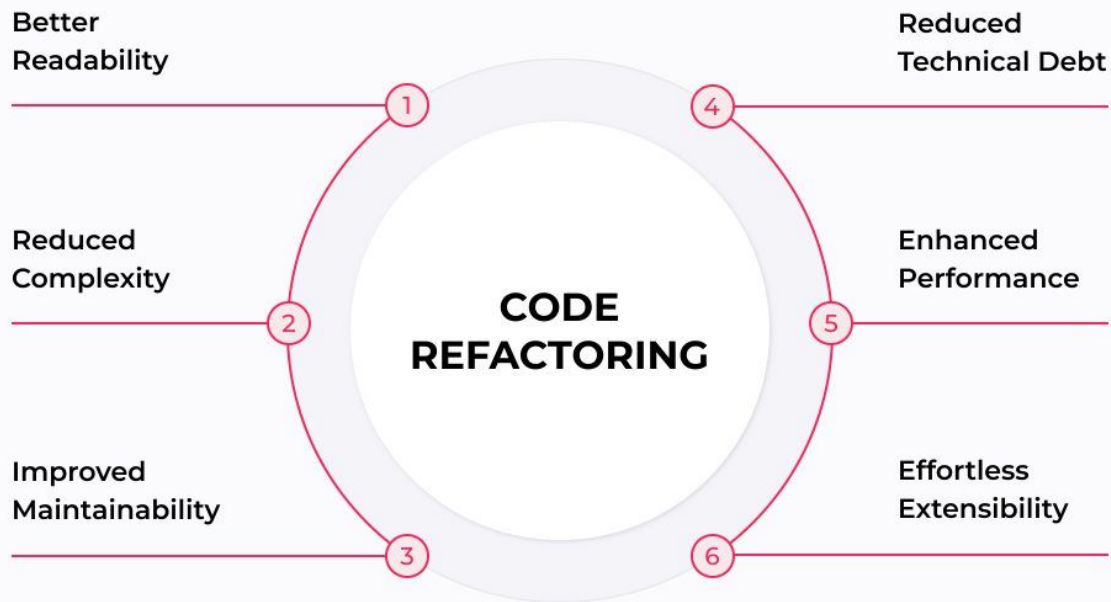
DD

Rendi il tuo programma versatile usando i dati. Rendi i dati unici. Separa i dati statici dai dati dinamici.

DRY, SSOT, SRP, DTA/LoD

Non ti ripetere, dividi bene le responsabilità pensando: chi è che deve gestire questo?

TAKEAWAY



GRAZIE!
Q/A



MORENO LOVATO

mlovato@runeheads.com

Discord: [morenolovatoruneheads](#)