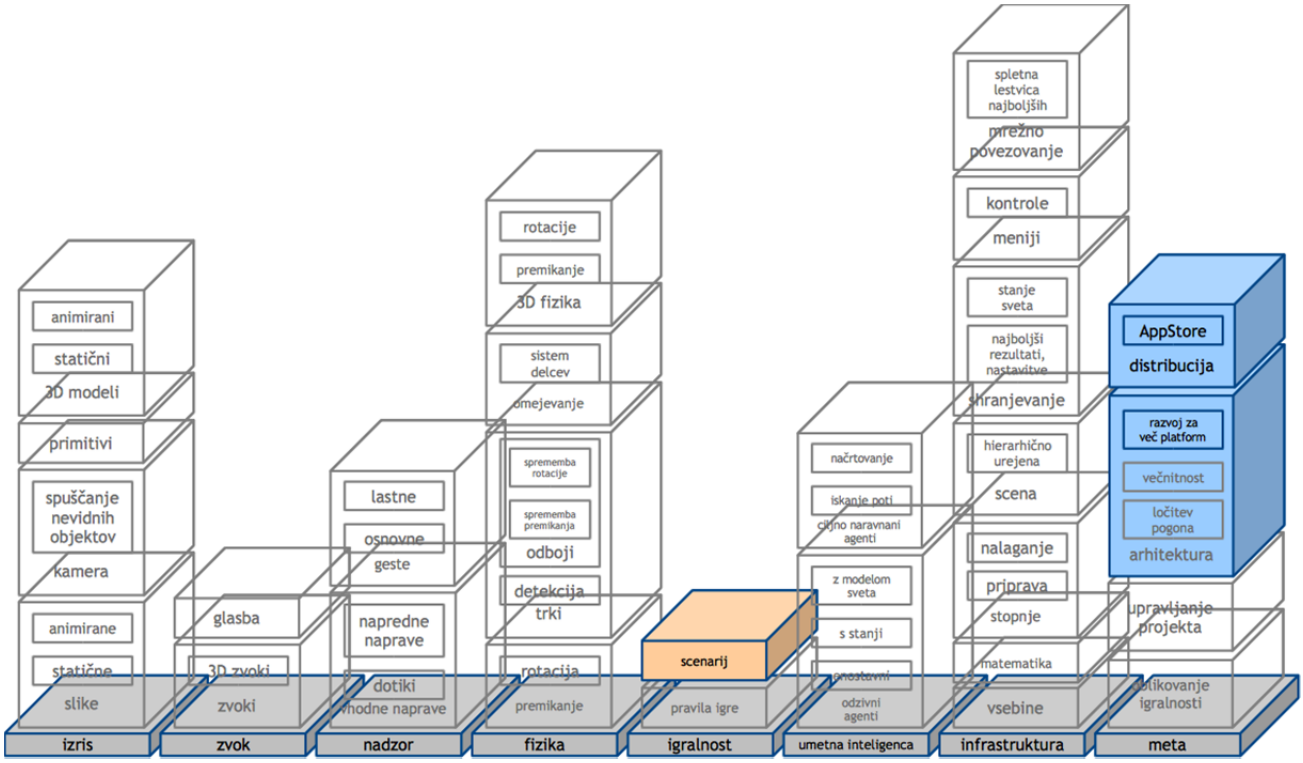
# Sklop 9


animirani
statični
3D modeli
primitivi
spuščanje nevidnih objektov
kamera
animirane
statične
slike
izris
glasba
3D zvoki
zvoki
zvok
lastne
osnovne
geste
napredne naprave
dotiki
vhodne naprave
nadzor
rotacije
premikanje
3D fizika
sistem delcev
omejevanje
sprememba rotacije
sprememba premikanja
odboji
detekcija trki
rotacija
premikanje
fizika
scenarij
pravila igre
igralnost
načrtovanje
iskanje poti
cilj no naravnani agenti
z modelom sveta
s stanji
odzivni agenti
umetna inteligenca
spletna lestvica najboljših
mrežno povezovanje
kontrole
meniji
stanje sveta
najboljši rezultati, nastavitve
shranjevanje
hierarhično urejena
scena
nalaganje
priprava
stopnje
matematika
vsebine
infrastruktura
AppStore
distribucija
razvoj za več platform
večnitnost
ločitev pogona
arhitektura
upravljanje projekta
oblikovanje igralnosti
meta

# Scenarij

S scenarijem mislimo na spreminjanje sveta igre, s čimer dosežemo odvijanje zgodbe ali tudi kar izvedbo določenih pravil igre. Podobno kot pri pravilih so rešitve zelo specifične glede na potrebe igre.

## Sekvence

Najbolj pogosta oblika izvajanja scenarija so uvodne in odjavne sekvence (lahko se pojavijo tudi sredi igranja). Podobno kot smo celotno igro razdelili na več stanj (običajno menijev), tako lahko potek samega igranja razdelimo na več notranjih stanj. Vzemimo za primer dirkačino, kjer v uvodu na ekran zapovrstjo pride več avtomobilov. Dogajanje razdelimo v najmanjša možna stanja in pripravimo enumeracijo:

```
typedef enum {
        IntroModeCar1Approach,
        IntroModeCar1Brake,
        IntroModeCar2Approach,
        IntroModeCar2Brake,
        IntroModeCar3Approach,
        IntroModeExplosion,
        IntroModeFire
} IntroMode;
```

Zdaj spet pripravimo avtomat stanj. Vsako stanje potrebuje pogoj, kdaj se nadaljuje v naslednje stanje (in katero). Eno izmed stanj mora biti začetno. Ob vstopu v stanje se zgodijo ustrezne spremembe (recimo avtomobilu nastavimo hitrost ali ukažemo zaustavljanje) nakar običajna pravila igre (fizikalni pogon) vodijo dogajanje dokler ni izpolnjen pogoj za napredovanje iz trenutnega stanja.

Potrebujemo torej spremenljivko za ternutno stanje:

```
IntroMode mode;
```

Na začetku ga nastavimo na prvo vrednost in nastavimo izhodiščne pogoje sekvence:

```
mode = IntroModeCar1Approach;

car = (Car*)[scene itemAtIndex:4];
car.velocity.y = -100;
...
```

V samem update koraku zanke izvajamo preverjanje pogojev trenutnega stanja in po potrebi napredujemo v naslednjega:

```
switch (mode) {
    case IntroModeCar1Approach:
        car = (Car*)[scene itemAtIndex:4];
        if (car.position.y < 420) {
            mode = IntroModeCar1Brake;
        }
        break;
    case IntroModeCar1Brake:
        car = (Car*)[scene itemAtIndex:4];
        car.velocity.y += dt * 100;
        if (car.velocity.y > 0) {
            car.velocity.y = 0;
            mode = IntroModeCar2Approach;
            car = (Car*)[scene itemAtIndex:5];
            car.velocity.y = -150;
        }
        break;
    case IntroModeCar2Approach:
        car = (Car*)[scene itemAtIndex:5];
        if (car.position.y < 450) {
            mode = IntroModeCar2Brake;
        }
        break;
    ...
    default:
        break;
}
```

Tako se bodo ena za drugo odvile spremembe na sceni, igralcu pa se bo pred očmi odvila želena sekvenca.

S takimi notranjimi stanji lahko izdelamo napredno igralnost na nivoju stopnje (sekvence) in tudi samih objektov na sceni (npr. vrata, ki se odprejo šele po nekem času in v sobo spustijo nove sovražnike).

# Naloga: scenarij

Poleg pravil igre, ki določajo zakonitosti v našem svetu, se igralnost doseže še z pripovedniško zmožnostjo iger, ki je vzporedna knjigam in filmom. Razvoj zgodbe v igrah največkrat izdelamo z vnaprej pripravljenimi dogodki, ki se sprožijo ob določenem času. Seveda ni nujno, da take animacije po scenariju skrbijo za napredovanje po zgodbi. Včasih, kot v primeru uvodnih in odjavnih sekvenc, zgolj pripomorejo k boljšemu vzdušju igre. Hkrati ni nujno, da med izvajanjem scenarija igralec ne more ničesar početi – v primeru vesoljskih streljačin se prihajanje nasprotnikov izvaja v obliki scenarija, seveda globoko povezanega z oblikovanjem samih stopenj.

Za nalogo je potrebno izdelati kakršnokoli proženje dogodkov skozi čas. Običajno to naredimo tako, da dogajanje razdelimo na posamezna stanja, med katerimi potem (linearno) prehajamo. Pri prehodu v novo stanje vzpostavimo potrebne začetne vrednosti, med izvajanjem stanja pa pregledujemo, če je izpolnjen pogoj za napredovanje v naslednje stanje. Torej v ozadju zopet stoji preprost končni avtomat.