

# Peliohjelmoinnin alkeet Unitylla

Valintarakenteet

# Sisällysluettelo

- Valintarakenne
- Vertailuoperaattorit
- Loogiset operaattorit

# Valintarakenne

- Unityssa jos halutaan reagoida johonkin tilanteeseen, niin käytetään valintarakenteita
  - Jos joku tapahtuma, toimitaan tällä tavalla
  - Jos meihin osuu vihollisen isku, vähennä elämäpisteitä
  - Jos elämäpisteet menevät loppuun, päätä peli
  - Jos hahmo osuu kerättävään esineeseen, kerää se esine
  - Jne jne.
- Koodissa tätä jossittelua tehdään if-lauseella

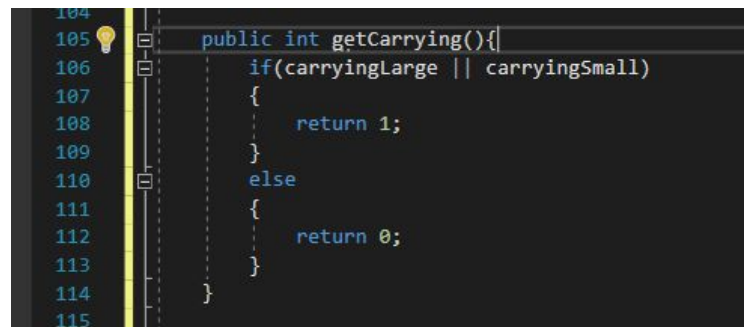
```
if (meihin osuu vihollisen isku) {  
    vähennä elämäpisteitä  
}
```

- Seuraavilla dioilla lisää if-lauseen rakenteesta

# Valintarakenne if-else

- if - else

```
if (ehto) {  
    1. koodilohko  
} else {  
    2. koodilohko  
}
```



A screenshot of a code editor showing a Java method `public int getCarrying(){}{`. The method contains an `if`-`else` statement. The `if` condition is `if(carryingLarge || carryingSmall)`. If true, it returns 1. If false, it returns 0. The code is color-coded: keywords in blue, literals in green, and identifiers in white. A yellow lightbulb icon is visible next to line 105.

```
104  
105 public int getCarrying(){}  
106     if(carryingLarge || carryingSmall)  
107     {  
108         return 1;  
109     }  
110     else  
111     {  
112         return 0;  
113     }  
114 }  
115
```

- Jos **ehto** toteutuu, niin suoritetaan **1. koodilohko**, muussa tapauksessa suoritetaan **2. koodilohko**

# Valintarakenne if-else if

- if - else if - else

```
if (ehto) {  
    1. koodilohko  
} else if (ehto2) {  
    2. koodilohko  
} else {  
    3. koodilohko  
}
```

- Jos **ehto** toteutuu niin suoritetaan pelkästään **1. koodilohko**
- Jos **ehto** ei toteudu niin tarkistetaan toteutuuko **ehto2** ja sen perusteella suoritetaan **2. koodilohko**
- Jos kumpikaan **ehdoista** ei toteudu niin suoritetaan **3. koodilohko**
- else if -lauseita voi olla niin monta kuin on tarpeen
- Tasan yksi koodilohkoista suoritetaan

```
125  
126  
127 public void checkWeight(GameObject item){  
128     if (item.CompareTag ("small")) {  
129         setCarrying (false, true);  
130     } else if (item.CompareTag ("large")) {  
131         setCarrying (true, false);  
132     } else {  
133         setCarrying (false,false);  
134     }  
135 }  
136  
137
```

# Valintarakenne if-if

- if - if - else

```
if (ehto) {  
    1. koodilohko  
}  
if (ehto2) {  
    2. koodilohko  
} else {  
    3. koodilohko  
}
```

- Jos **ehto** toteutuu niin suoritetaan  
**1. koodilohko**
- Jos **ehto2** toteutuu niin suoritetaan  
**2. koodilohko**
- Jos **ehto2** ei toteudu niin suoritetaan  
**3. koodilohko**
- if-lauseet ovat irrallisia ja kummatkin tarkistetaan riippumatta toisesta
- Ensimmäinen if-lause ei liity else-lauseeseen

# Vertailuoperaattorit

- Varsin usein ehtolauseissa verrataan eri lukuja keskenään
  - Onko toinen luku isompi kuin toinen? Ovatko luvut yhtä suuret?
- Näitä varten ovat ala-asteeltakin osittain tutut vertailuoperaattorit

Operaattori	Merkintä
Yhtäsuuruus	==
Erisuuruus	!=
Suurempi kuin	>
Suurempi kuin tai yhtä suuri	>=
Pienempi kuin	<
Pienempi kuin tai yhtä pieni	<=

Huom.

- Yhtä yhtäsuuruusmerkkiä käytetään asettamaan arvo
  - $x = 10$
  - $x$  saa arvon 10
- Kahta yhtäsuuruusmerkkiä käytetään vertailemaan arvoja
  - $x == 10$
  - onko  $x$ :n arvo 10?

# Vertailuoperaattori - esimerkki

```
if (3 < 5) {  
    //ehto toteutuu  
}  
  
if (6 < 5) {  
    //ehto ei toteudu  
}  
  
if (5 > 5) {  
    //ehto ei toteudu  
}  
  
if (5 >= 5) {  
    //ehto toteutuu  
}
```

```
if (1 == 1) {  
    //ehto toteutuu  
}  
  
if (1 != 1) {  
    //ehto ei toteudu  
}
```



# Loogiset operaattorit

- Loogisilla operaattoreilla voidaan yhdistää useampi ehto yhdeksi totuusarvoksi
  - Esimerkiksi: Menen kauppaan jos paistaa aurinko **JA** on lämmin
  - Menen kauppaan jos paistaa aurinko **TAI** on lämmin
  - Menen kauppaan jos **EI** sada vettä
- C#:ssa (ja monessa muussakin kielessä) loogiset operaattorit toimivat näillä merkeillä:

Operaattori	Merkintä
AND	&&
OR	
Negaatio	!

- Negaatio siis kääntää totuusarvon päinvastaiseksi
  - !true -> false
- OR-merkki on siis kaksi pystyviivaa
  - Pystyviiva löytyy näppäimistöltä Z-kirjaimen vasemmalta puolelta kun painaa AltGr

# Loogiset operaattorit - esimerkkejä

```
if (true && true) {  
    //ehto toteutuu  
}
```

```
if (true || false) {  
    //ehto toteutuu  
}
```

```
if (!false) {  
    //ehto toteutuu  
}
```

```
if (true && false) {  
    //ehto ei toteudu  
}
```

```
if (false || false) {  
    //ehto ei toteudu  
}
```

```
if (!true) {  
    //ehto ei toteudu  
}
```

true ja false voivat siis  
olla mitä tahansa  
ehtoja, esimerkiksi:  
1 < 2 (true)  
2 == 3 (false)

# Loogiset operaattorit - esimerkki

```
if (true && true && true && false && true) {  
    //ehto ei toteudu  
    //AND-operaattori vaatii että kaikki ehdot toteutuvat  
}  
  
if (false || true || true || false || false) {  
    //ehto toteutuu  
    //OR-operaattorille riittää että yksikin ehdoista toteutuu  
}  
  
if (!!true) {  
    //ehto toteutuu  
    //vertaa matematiikassa  $-(-1) == 1$   
}
```