IN00BQ93 Laite- ja tuotesuunnittelun syventävät opinnot: ohjelmoinnin jatkokurssi

Qt ympäristön lataaminen ja asennus

1: Mene selaimella sivulle http://www.qt.io/

2: Klikkaa sivulla oikeassa yläkulmassa olevaa linkkiä, jossa teksti **Download. Try. Buy.**

3: Aukeavalla sivulla klikkaa **Open Source** osasta löytyvää painiketta **Go open source**

4: Klikkaa aukeavalla sivulla painiketta **Download**, jonka jälkeen **Qt Online installer** tiedosto latautuu koneellesi.

5: Kun tiedosto ladattu, niin suorita tiedosto. Jos **Windows** heittää jotain herjaa, niin valitse **Run anyway** tms.

6: Klikkaa asennuksen ensimmäisessä ikkunassa (Welcome to the Qt online installer) painiketta **Next**

7: Tässä vaiheessa ei rekisteröidytä, joten klikkaa ikkunassa painiketta **Skip**, ja sen jälkeen avautuvassa ikkunassa klikkaa **Next**

8: Asenna Qt siihen hakemistoon (C:\Qt) jota asennusohjelma ehdottaa. Varmista, että kohdassa **Assosiate common file types with Qt Creator** on rasti, ja klikkaa **Next**

Asennusohjeet jatkuvat seuraavalla sivulla!

IN00BQ93 Laite- ja tuotesuunnittelun syventävät opinnot: ohjelmoinnin jatkokurs

Qt ympäristön lataaminen ja asennus

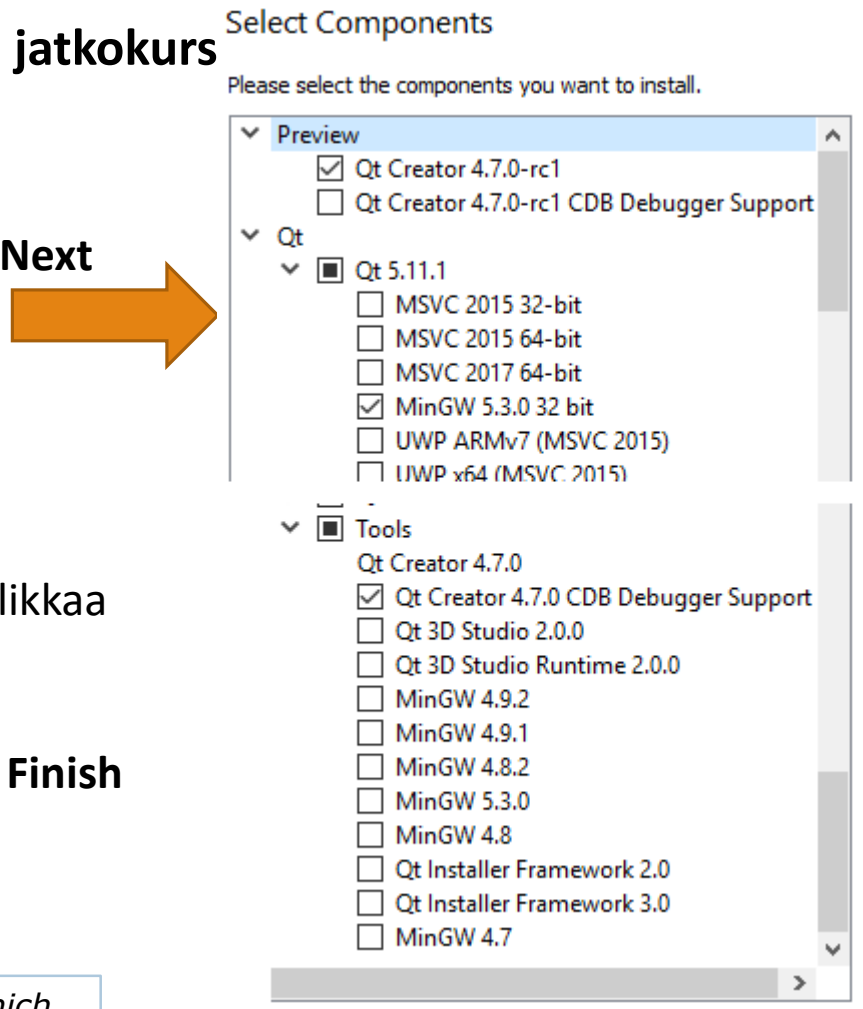
9: Ikkunassa **Select Components** valitse oikealla kuvassa olevat valinnat ja klikkaa **Next**

- Kohdassa **Tools** voi olla jo oletusvalintana Microsoft Console Debugger CDB

10: Ikkunassa **License Agreement** lue ja hyväksy ehdot, ja klikkaa **Next**

11: **Start Menu shortcuts** ikkunassa klikkaa **Next** ja sitten seuraavassa ikkunassa klikkaa **Install** painiketta, jonka jälkeen Qt ohjelmointiympäristö asennetaan koneeseesi.

12: Ikkunassa **Completing …** anna olla rasti kohdassa **Launch Qt Creator** ja klikkaa **Finish** painiketta, jolloin **QtCreator** työkalun pitäisi käynnistyä.



Qt Creator is a cross-platform C++, JavaScript and QML integrated development environment which is part of the SDK for the Qt GUI Application development framework. It includes a visual debugger and an integrated GUI layout and forms designer. The editor's features include syntax highlighting and autocompletion. Qt Creator uses the C++ compiler from the GNU Compiler Collection on Linux and FreeBSD. On Windows it can use MinGW or MSVC with the default install and can also use Microsoft Console Debugger when compiled from source code.

Qt ympäristön testaus ja Qt Creator kehitystyökalun käyttöönotto

1: Valitse Qt Creator työkalun päävalikosta **File->New File or Project ...**

2: Valitse projektiksi **Non-Qt Project->Plain C++ Application** ja klikkaa **Choose**

3: Anne projektille nimeksi **MyFirstQtProject** ja valitse kansio (esim. C:\data) ja klikkaa **Next**

4: Ikkunassa **Define Build System** anna olla valittuna **qmake** ja klikkaa **Next**

- Lisää tietoa osoitteesta **http://doc.qt.io/qtcreator/creator-project-other.html**

5: **Kit Selection** ikkunassa valitse **Desktop Qt 5.11.1 MinGW 32bit** ja klikkaa **Next**

- Lisää tietoa osoitteesta **http://doc.qt.io/qtcreator/creator-build-settings.html**

6: Lopuksi klikkaa painiketta **Finish**, niin projekti luodaan ja työkaluun aukeaa **main.cpp** tiedosto.

7: Avaa resurssienhallintaohjelmalla hakemisto **C:\data\MyFirstQtProject** (projekti luotu tänne). Tämä hakemisto sisältää projektin tiedostot esim. **main.cpp**

- Käydään tunneilla tarvittaessa läpi ohjatusti työkalun toimintoja.

- Poista **main.cpp** tiedostossa oleva koodi ja kirjoita alla oleva koodi **main.cpp** tiedostoon

```
#include <iostream>

int main()
{
    std::cout << "Qt: Plain C++ Project" << std::endl;

    return 0;
}
```

- Klikkaa editorin alaosassa olevaa painiketta **4: Compile output**, niin esille tulee ikkuna joka näyttää miten käännösprosessi etenee.
- Koodin kääntäminen ja linkittäminen (=build) suoritetaan valitsemalla päävalikosta kohta **Build->Build Project "..."** tai sen voi myös tehdä klikkaamalla vasemmassa alakulmassa olevaa **vasaraa. *Suorita Build!***

- Kun **Build** toiminto on onnistuneesti suoritettu ensimmäisen kerran, niin sinne kansioon minne projekti luotiin tulee **alikansio** nimellä **build-MyFirstQtProject-Desktop_Qt_...-Debug** jonka alle suoritettavat tiedostot tulevat.

- Ohjelma suoritetaan/ajetaan valitsemalla päävalikosta kohta **Build->Run** tai klikkaamalla vasemmassa alakulmassa olevaa vihreää kolmiota. **Älä klikkaa kolmiota jossa on kuoriainen päällä!**

- *Suorita ohjelmaa!*

- Poista ohjelman **return** lauseesta numero 0, ja buildaa ohjelma jolloin saat esille käännösvirheen. Käännösvirheet aukeavat **Issues** ikkunaan. Virheellisen koodirivin kohdalle ilmestyy punaisella pohjalla huutomerkki virheen merkiksi.

Edellisen esimerkin lyhyt läpikäynti

- Edellisessä esimerkissä rivillä 1: ***#include <iostream>*** sisällytetään ohjelmaan kirjastotiedosto **iostream.h**. Kääntäjän kannalta tiedosto on ikään kuin kirjoitettu suoraan lähdekoodiin juuri ohjelman alkuun.

- Aina, kun ajat kääntäjän, se kutsuu ensin toista ohjelmaa eli esikäsittelijää. Esikäsittelijää ei tarvitse kutsua itse suoraan, vaan sitä kutsutaan automaattisesti joka kerran kääntäjää ajettaessa. Esikäsittelijän tehtävänä on käydä läpi koodi rivi riviltä ja etsiä ne rivit, jotka alkavat #-merkillä. Kun esikäsittelijä havaitsee tuollaisen rivin, se alkaa muokata koodia. Muokattu koodi luovutetaan sitten kääntäjälle.

- Komento **include** on esikäsittelijän komento, joka sanoo "Seuraavana on tiedoston nimi. Hae tuo tiedosto ja lue se tähän". Kulmasulkumerkit tiedoston nimen molemmin puolin kehottavat esikäsittelijää "hakemaan tiedostoa kaikista yleisistä paikoista". Jos kääntäjän asetukset ovat oikein, esikäsittelijä hakee tiedostoa hakemistosta, jossa säilytetään kaikkia .h tiedostoja.

- Tiedosto **iostream.h** (input/output stream) on **cout-olion** käytössä silloin, kun se tulostaa tekstiä näytölle. Rivin 1 ansiosta tiedosto **iostream.h** sijoitetaan koodiin aivan kuin olisit kirjoittanut sen itse sinne. Kääntäjä näkee sitten koodin paikallaan jatkaessaan kääntämistä.

- Rivi 2 aloittaa todellisen ohjelman funktiolla main(). Jokaisessa C++ - ohjelmassa on main()-funktio. Funktio on koodilohko, joka suorittaa jonkun toiminnon. Funktiot toteuttavat tehtävänsä toisten funktioiden kutsumina, mutta main()-funktio on ainutlaatuinen. Kun ohjelma käynnistetään, kutsutaan main()-funktiota automaattisesti. Kaikkien funktioiden, niin myös main()-funktion, on määritettävä, millaisen arvon se palauttaa.

Std -nimiavaruus

- ISO C++ -standardi määrittelee omaan käyttönsä **std**-nimisen nimiavaruuden.
- Tämän nimen alle on kerätty kaikki kielen määrittelyn esittelemien rakenteiden nimet (muutamaa poikkeusta, kuten funktiota **exit** lukuun ottamatta).
- Esimerkiksi C:n tulostusrutiini **printf** ja C++ :n tulostusolio **cout** ovat ISO C++ :n mukaisesti toteutetussa kääntäjässä nimillä **std::printf** ja **std::cout** , jotta ne eivät aiheuttaisi ongelmia ohjelman muiden nimien kanssa.
- Koska kirjastoon kuuluu myös C -kielestä tulleita funktioita, std -nimiavaruus sisältää satoja nimiä. Std -nimiavaruus on olemassa kaikissa nykyaikaisissa C++ -kääntäjissä, ja ainoastaan sen rakenteiden käyttäminen on sallittua – tähän nimiavaruuteen ei saa itse lisätä uusia ohjelmarakenteita.
- Koko C++:n standardikirjasto on std-nimiavaruuden sisällä, ja monet lisäkirjastot käyttävät omia nimiavaruuksiaan.
- Seuraavassa esimerkissä otetaan **std** nimiavaruus käyttöön kommennolla **using namespace std;**

Std nimiavaruuden käyttöönotto

- Kommentoi/poista **main.cpp** koodi kokonaan ja lisää alla oleva koodi **main.cpp** tiedostoon, suorita Build ja aja ohjelmaa

```
#include <iostream>

using namespace std;

int main()
{
    cout << "Qt: Plain C++ Project" << endl;

    return 0;
}
```

IN00BQ93 Laite- ja tuotesuunnittelun syventävät opinnot: ohjelmoinnin jatkokurssi

Kertausta funktionaalisesta ohjelmoinnista

- Kommentoi **main.cpp** tiedostossa oleva koodi kokonaan, ja lisää alla oleva koodi tiedostoon. Suorita Build ja aja ohjelmaa.

```
#include <iostream>

using namespace std;

double summa(double parametriLukuYksi, double parametriLukuKaksi)
{
   double tulos = 0;
   tulos = parametriLukuYksi + parametriLukuKaksi;

   return tulos;
}

int main()
{
   double tulos, lukuYksi, lukuKaksi;
   lukuYksi = 10;
   lukuKaksi = 5;
   tulos = summa(lukuYksi, lukuKaksi);
   cout << lukuYksi << "+" << lukuKaksi << "=" << tulos << endl;

   return 0;
}
```

Tehtävä 1: Lisää ohjelmaan laskutoimitukset erotus, tulo ja osamäärä yllä olevan mukaisesti