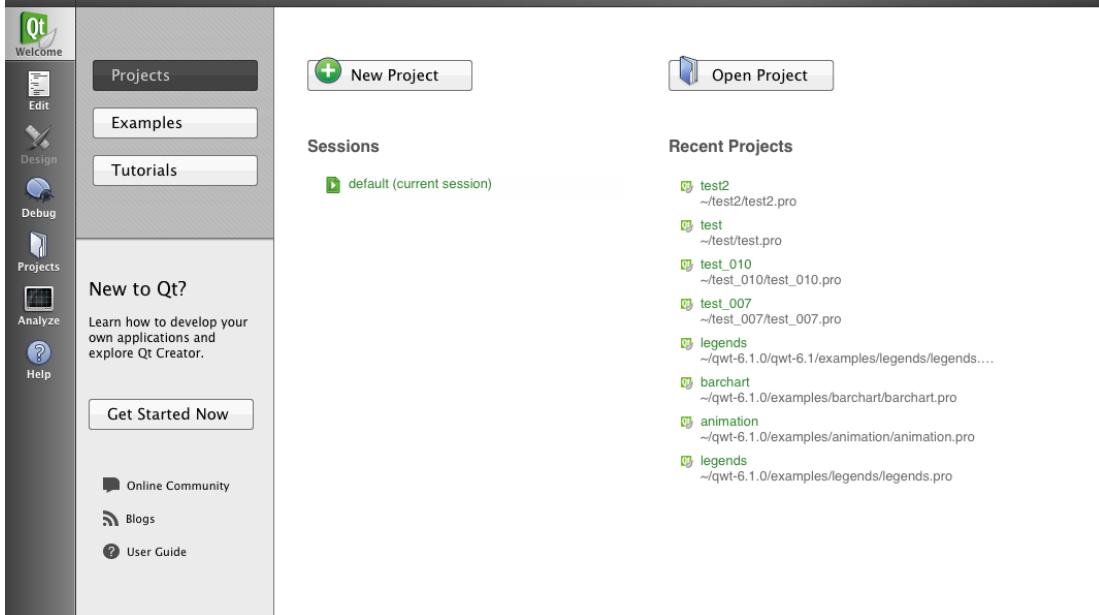


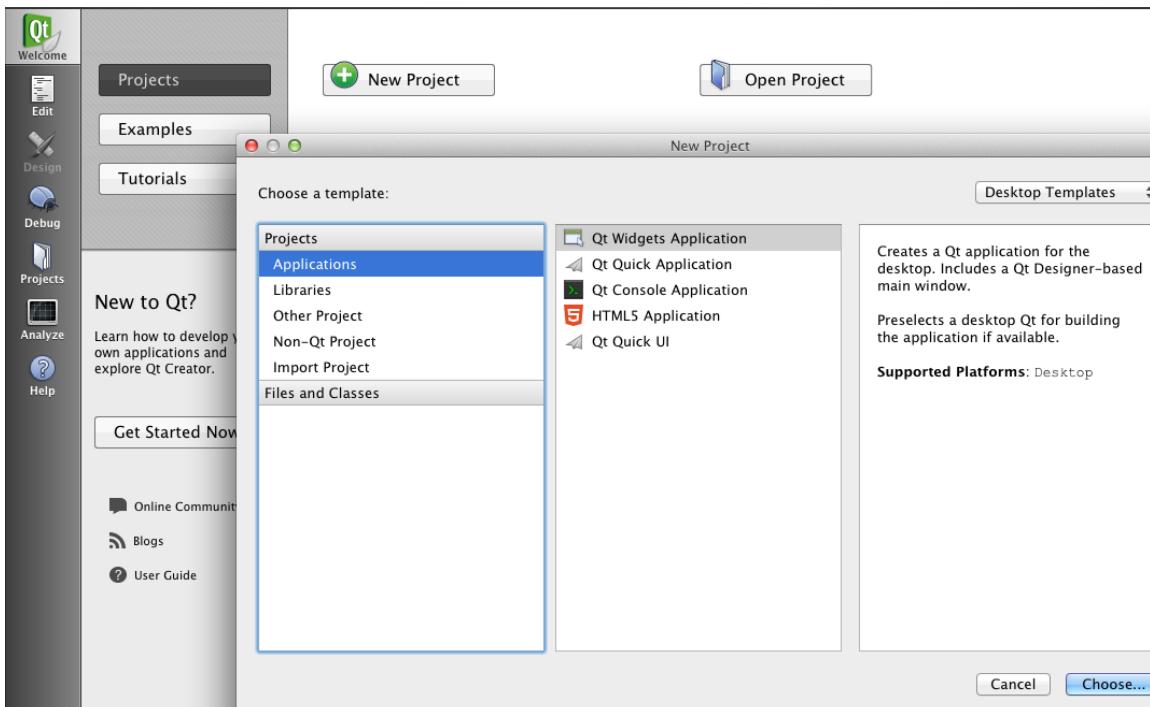
* QT 프로젝트 열기

QT 프로그래밍을 시작하기 위한 과정은 다음과 같다.

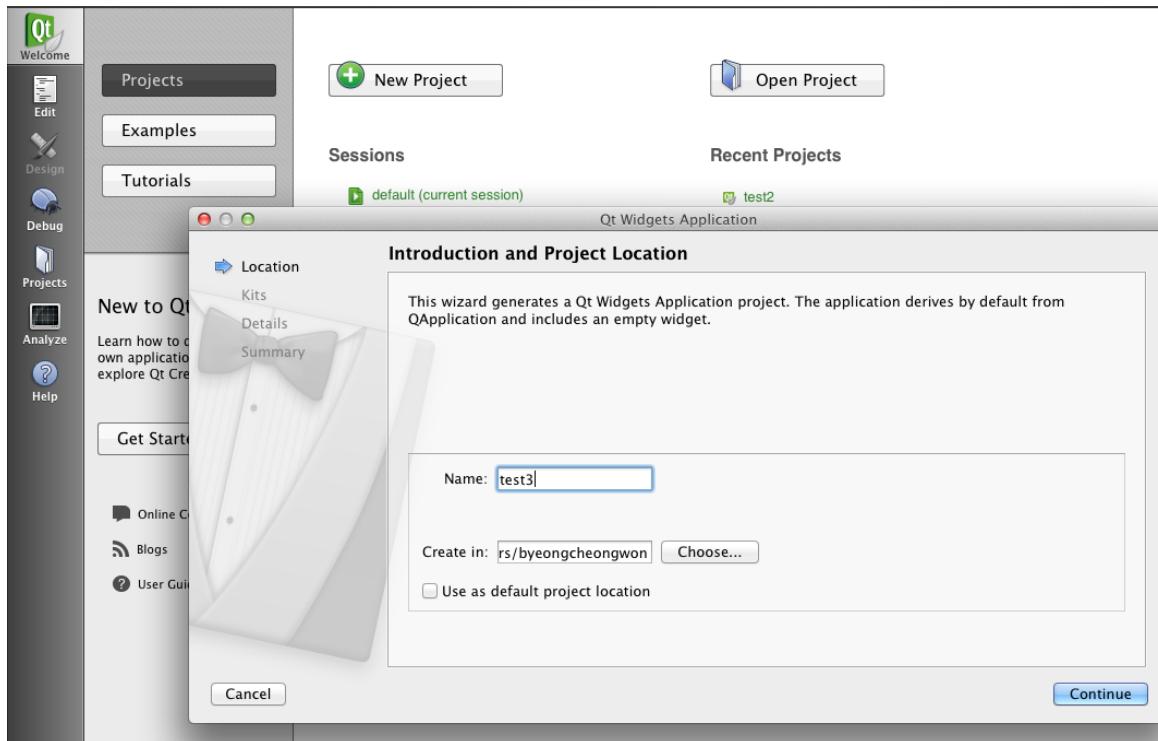
(1) New Project 클릭



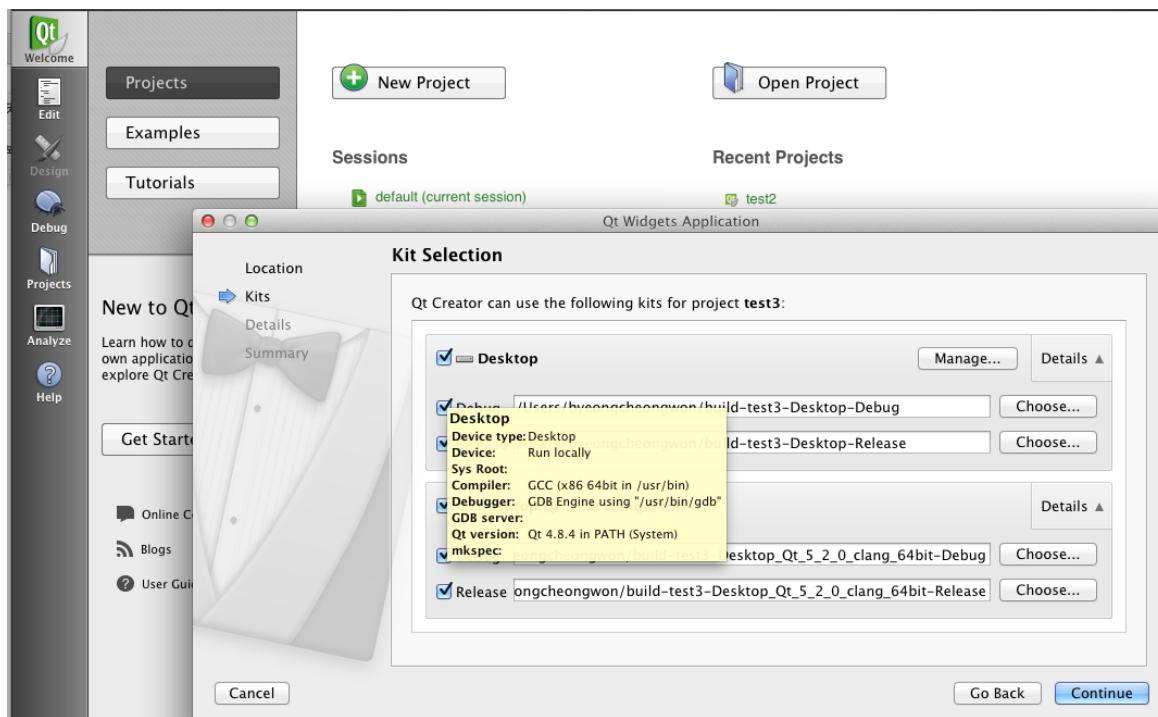
(2) Application 항목에서 Qt Widgets Application 클릭 choose 버튼 클릭



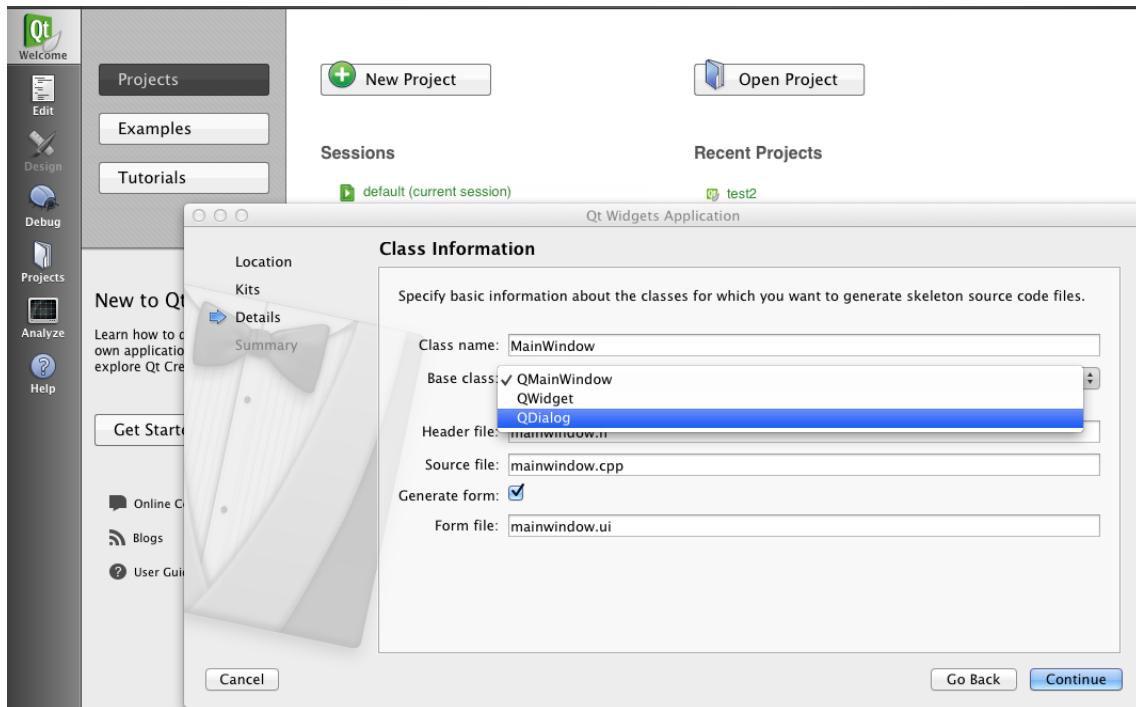
(3) 프로젝트 이름 정하고 쓰기



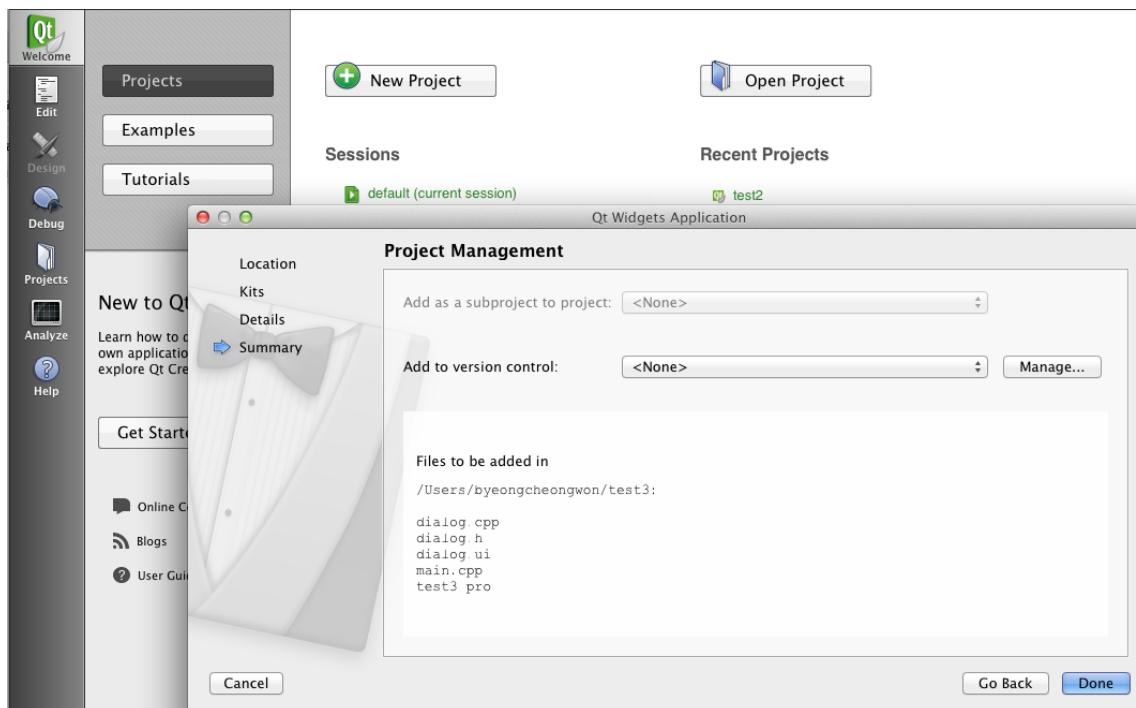
(4) continue 버튼 클릭



(5) Base class 탭에서 QDialog 클릭하고 Continue 클릭

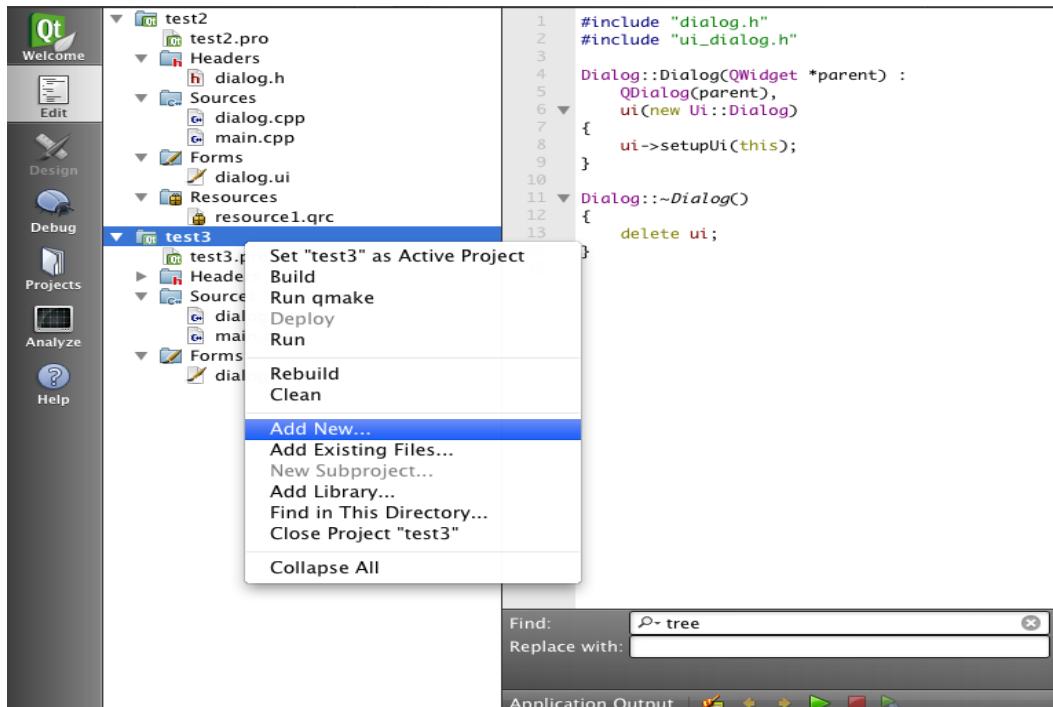


(6) Done 버튼 클릭

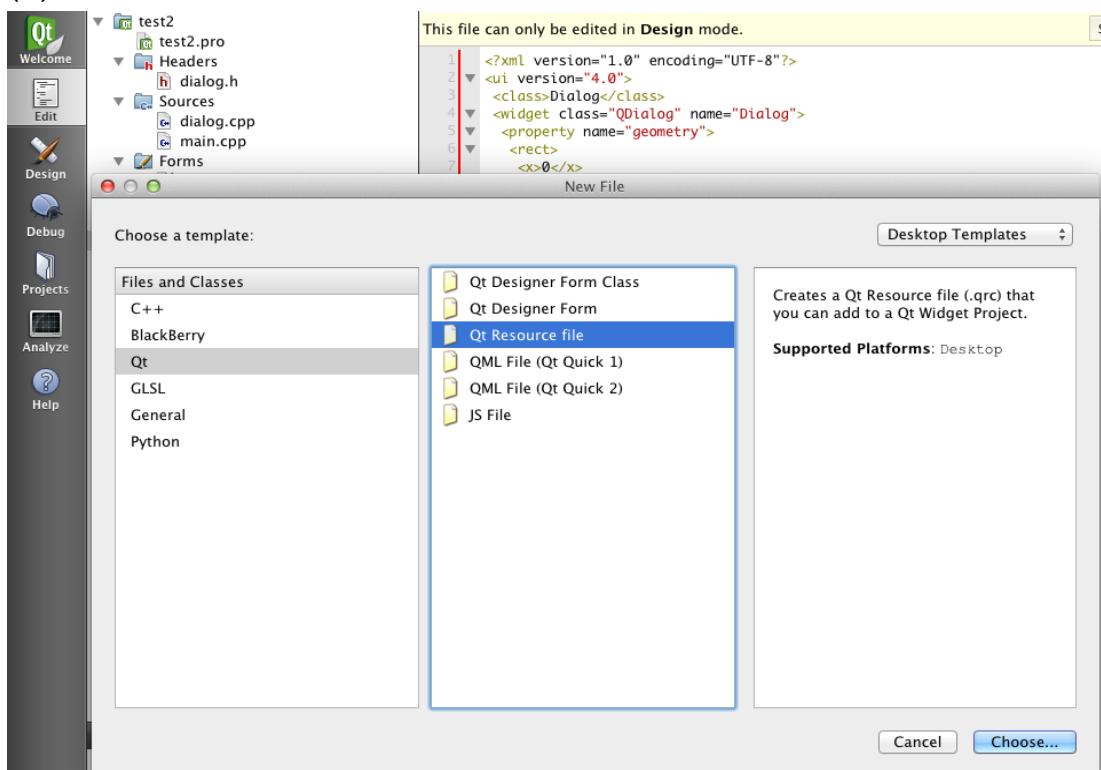


* Resource 폴더 열기

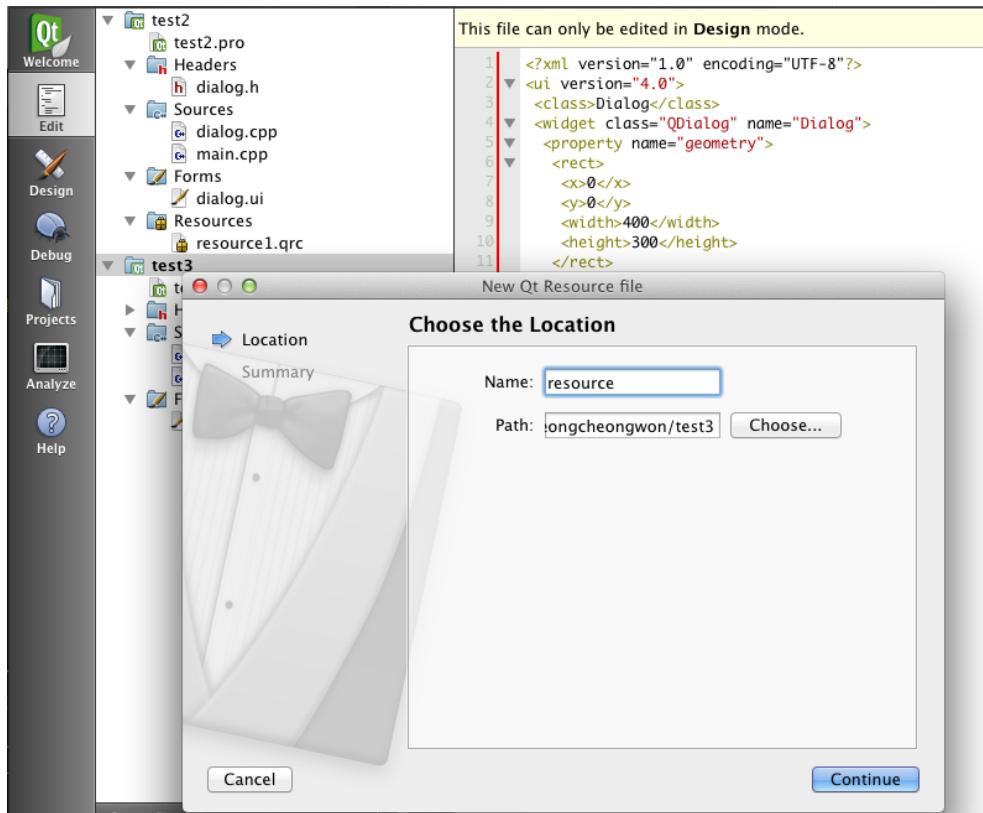
(1) 해당 프로젝트에서 아래 화면과 같이 Add new 클릭



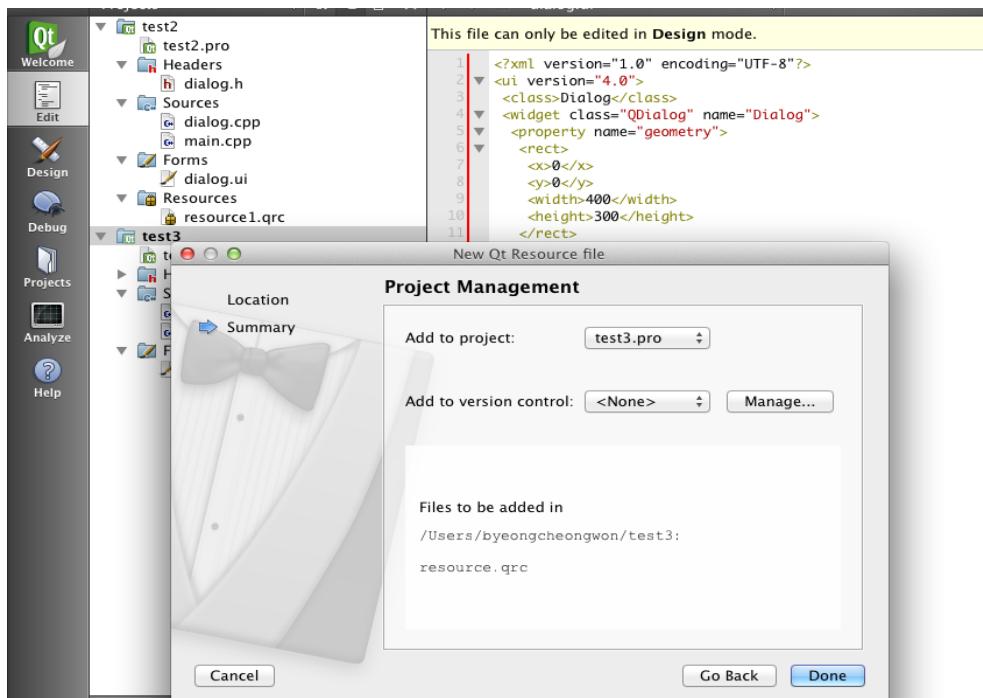
(2) Qt 항목에서 Qt Resource file 선택하고 Choose 버튼 클릭



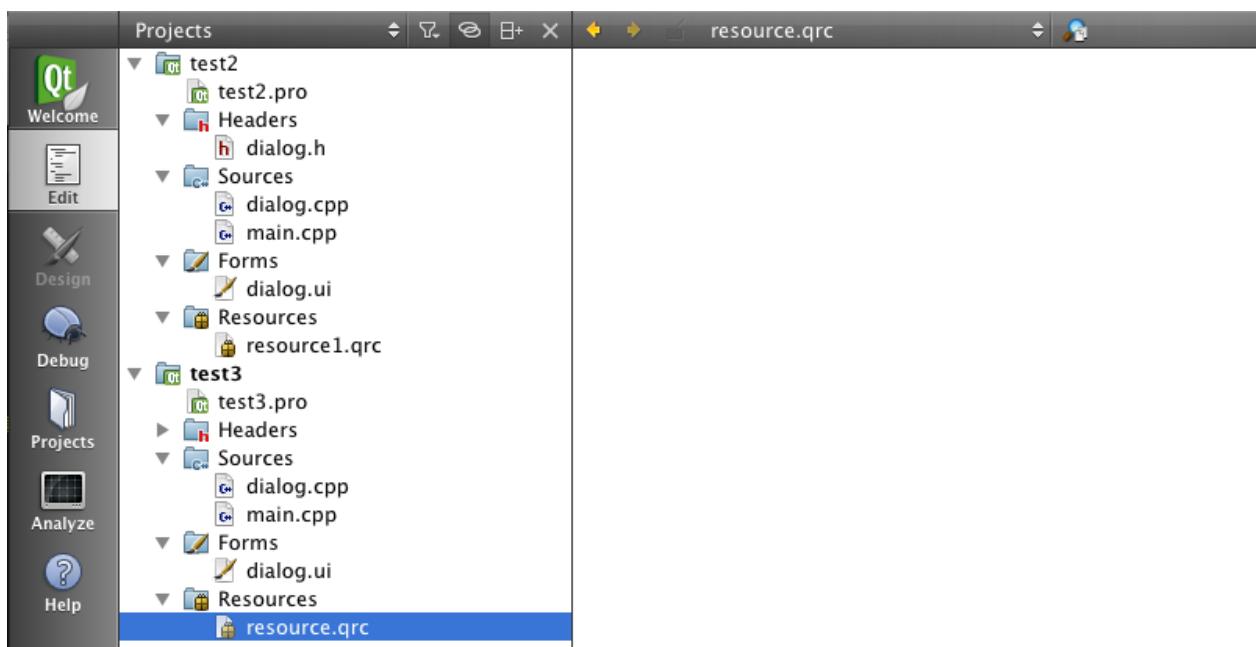
(3) Resource 폴더 이름 정하고 Continue 버튼 클릭



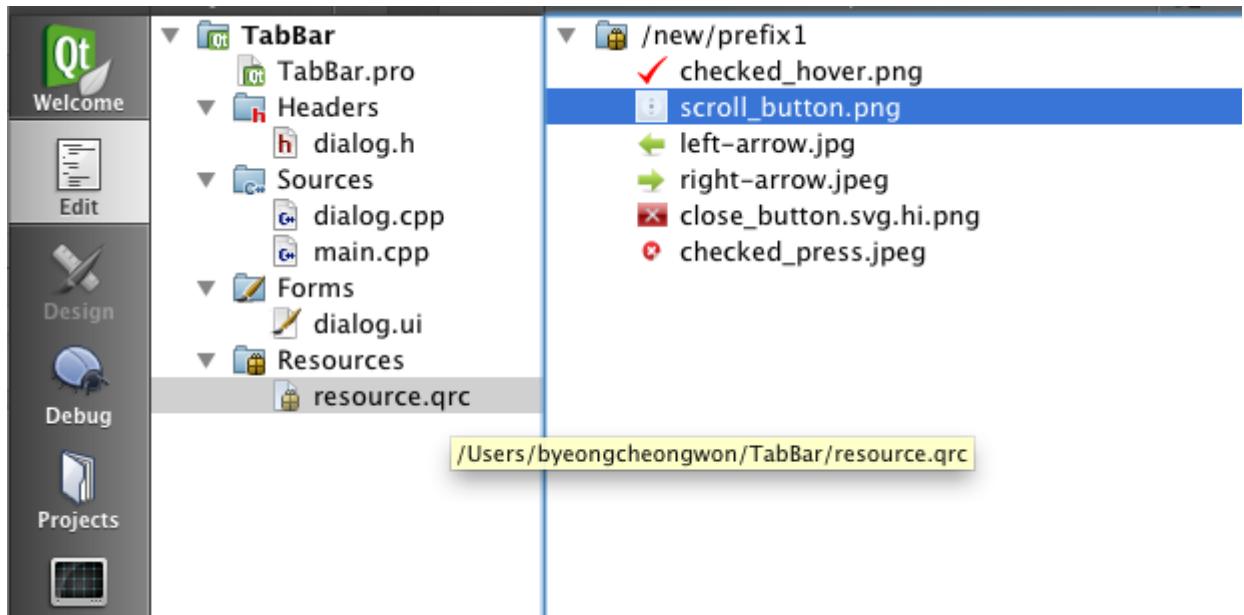
(4) Done 버튼 클릭



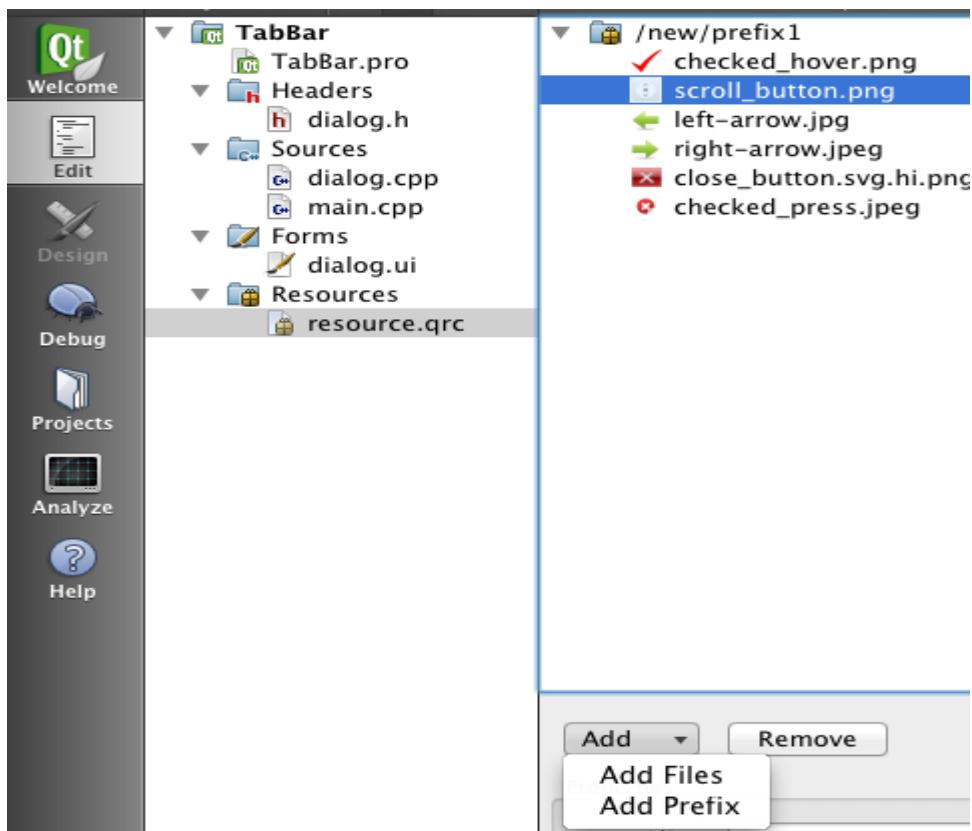
(5) Resource 파일 생성 완료



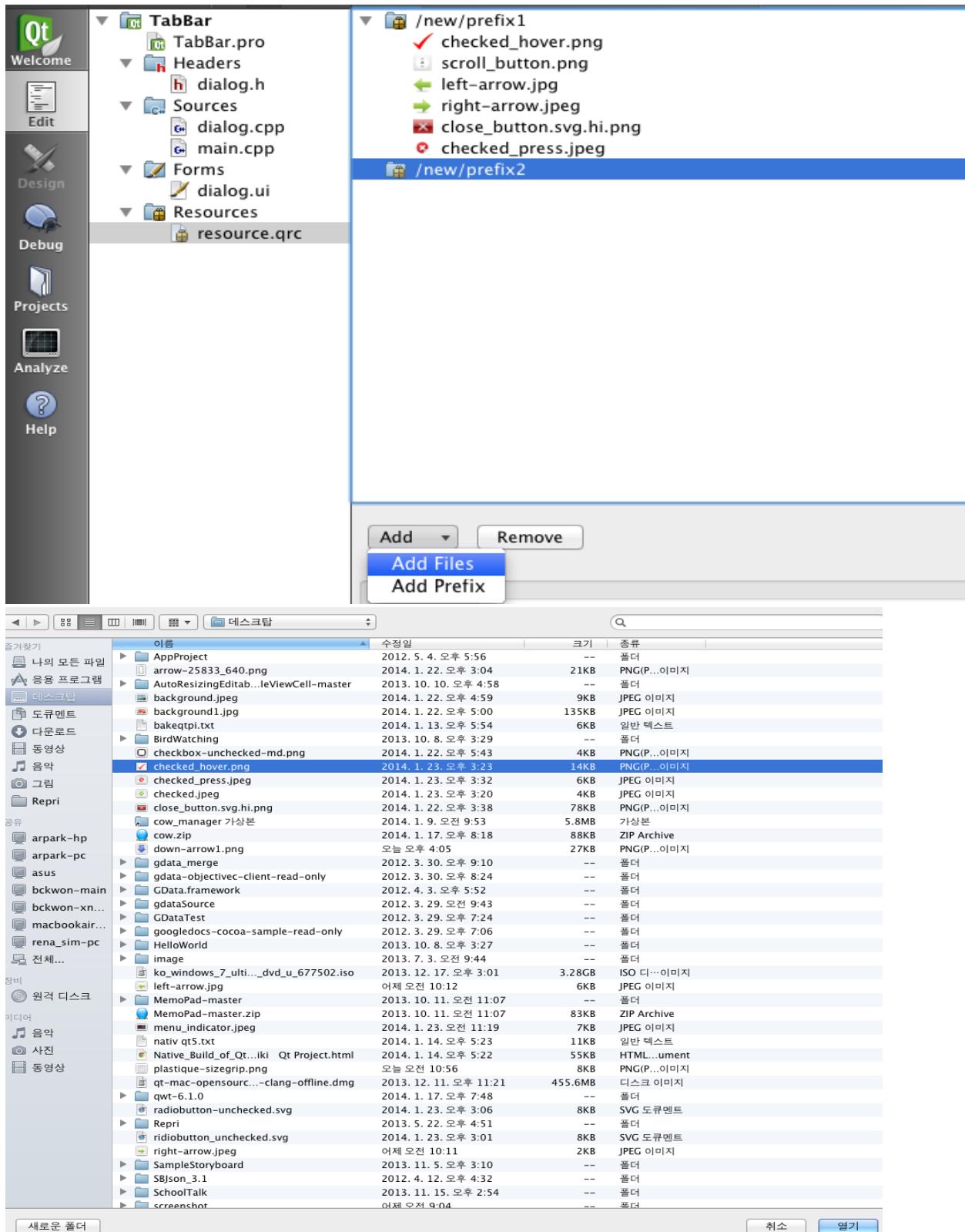
* Resource 폴더에 이미지(그림) 파일 넣기
(1) Resource 폴더를 클릭



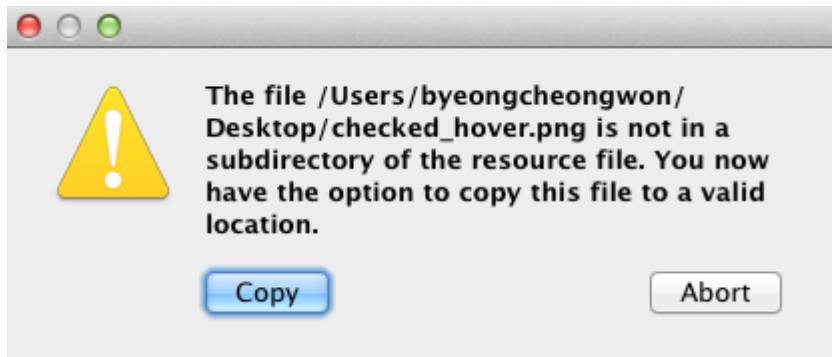
(2) 맨 아래에 있는 Add 버튼 클릭한 뒤 Add Prefix 클릭



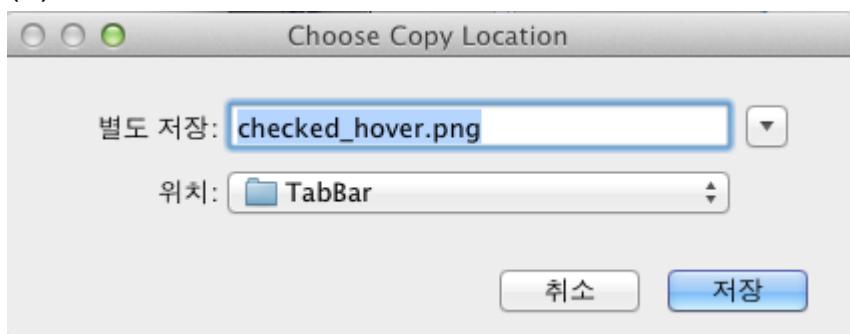
(3) 새로운 Prefix 안에서 Add Files를 누르고 이미지 추가



(4) copy 버튼 클릭



(5) 저장 버튼 클릭



(6) 마지막으로 Ctrl+S(저장) 함으로써 resource 폴더 안에 이미지 추가 완성

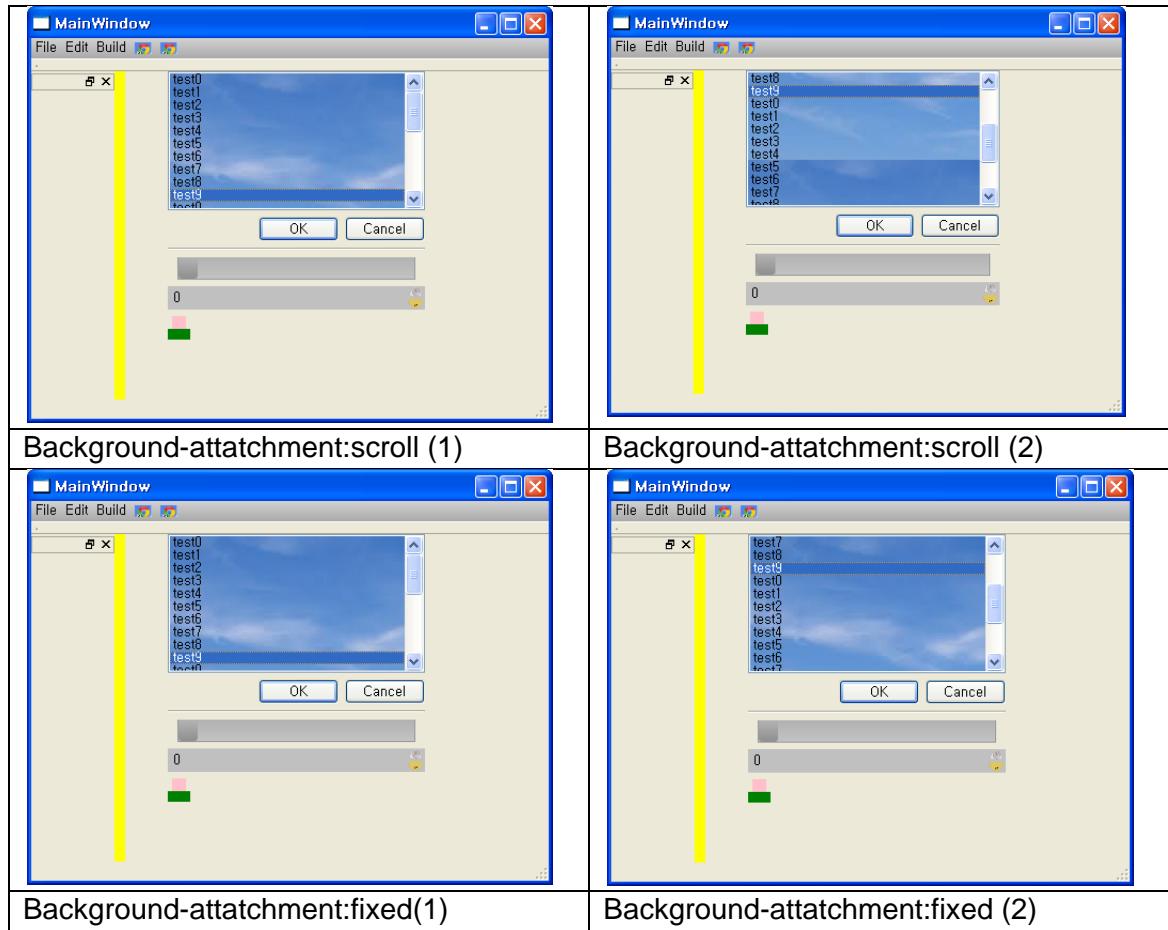


01. QAbstractScrollArea

- QTextEdit, QListWidget { background-color:white; background-image: url(:/new/prefix1/images.jpg); background-attachment: scroll; }

TextEdit이나 ListView에서 설정된 영역보다 내용이 많은 경우 스크롤바가 생기게 된다. 이 때 할 수 있는 설정이다.

일단 기본적으로 배경을 흰색으로 하고, 이미지로 하늘 배경 이미지를 가져왔다. 그리고 background-attachment속성에서 fixed와 scroll, local이 있는데 scroll과 local은 유사한 부분이 많다. 우선 scroll은 image 영역이 초과되면 qt에서는 이미지를 세로 바둑판식 처럼 새로 그리게 된다. fixed에서는 스크롤을 움직여도 그림은 고정된 채로 있다.



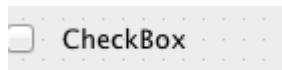
02. QCheckBox

CheckBox 란 GUI 환경에서 사용자에게 여러 항목 중 한 개 이상 선택하게 할 때 사용되는 작은 사각형이다.

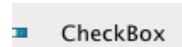
- **QCheckBox {spacing 5px;}** : spacing 을 통해서 버튼과 이름의 간격을 조절하는 것이다.



- **QCheckBox::indicator {width: 13px; height: 13px;}** : CheckBox 의 크기를 설정하는 것으로 가로와 세로의 길이를 설정한다.



- **QCheckBox::indicator:unchecked {image :url(~);}** : 이미지의 경로를 통해서 이미지를 체크버튼에 삽입하는 것으로, 여기서 **unchecked** 는 체크버튼을 누르지 않았다는 상태를 보여주는 것으로 체크버튼을 누르지 않으면 경로에 있는 이미지가 나오게 되고, 체크버튼을 눌렀을 때는 기존의 체크버튼 이미지가 보인다.



- **QCheckBox::indicator:unchecked:hover {image :url(~);}** : **unchecked** 는 체크버튼을 누르지 않았을 때의 상태를 나타내주는 것으로, 이 상태에서 **hover** 라는 명령을 해주면 마우스 포인터를 체크박스에 놓아두었을 시에 아래 그림과 같은 경로의 이미지가 보인다.



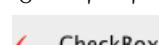
- **QCheckBox::indicator:unchecked:pressed {image :url(~);}** : **pressed** 는 **hover** 와 반대 개념으로 체크버튼을 눌렀을 때 아래 그림과 같은 경로의 이미지가 보인다.



- **QCheckBox::indicator:checked {image :url(~);}** : **checked** 는 체크버튼을 눌렀을 시에 아래 그림과 같은 경로의 이미지로 나타내 준다.



- **QCheckBox::indicator:checked:hover {image :url(~);}** : **hover** 는 4 번째에서 설명했던 것처럼 체크박스가 **check** 된 상태에서 마우스 포인터를 갖다 놓으면 아래 그림과 같은 경로의 이미지를 보여주게 된다.



- **QCheckBox::indicator:checked:pressed {image :url(~);}** : **pressed** 는 **hover** 와 반대 개념으로 체크박스가 **check** 된 상태에서 체크버튼을 클릭하고 있으면 아래 그림과 같은 경로를 통한 이미지를 보여주게 된다.

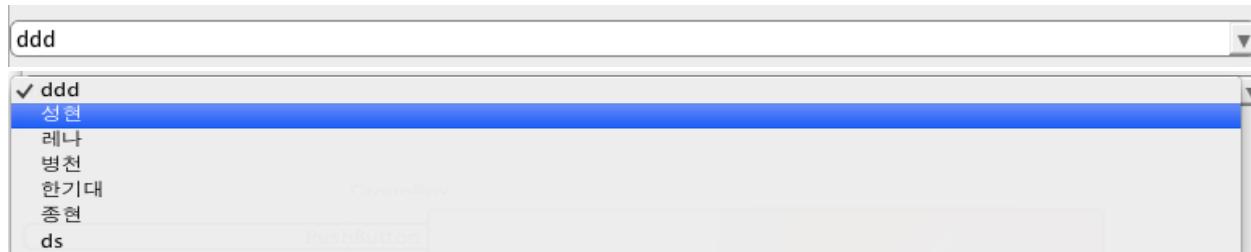


- QCheckBox::indicator:indeterminate:hover {image :url(~~);} :
- QCheckBox::indicator:unchecked:hover {image :url(~~);}

03. QComboBox

ComboBox 란 사용자가 직접 정보를 입력하거나 나열된 항목들 중에서 하나의 항목을 선택하여 정보를 입력할 수 있는 컨트롤이다.

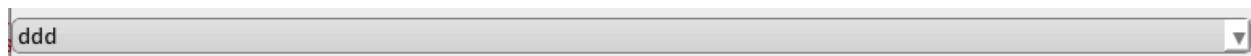
- QComboBox {border: 1px solid gray; border-radius: 5px; padding: 1px 18px 1px 3px; min-width: 6em;} : border 는 콤보박스의 바깥 선을 뜻하는 것으로, px 는 border 의 굵기, solid 는 border 의 종류, gray 는 border 의 색을 의미한다. 여기서 선의 종류인 solid 는 단선을 의미한다. 따라서 border: 1px solid gray 는 1px 크기이면서 solid 형태이고 회색인 선을 의미하는 것이다. border-radius 는 양 끝의 반원을 의미한다. 따라서 border-radius: 5px 는 양 끝을 5px 길이의 반지름을 가진 원으로 표현하라는 것과 같다. padding 은 콤보박스의 글자나 문자에 외곽에 빈 공간을 설정해준다. 마지막으로 min-width 는 가로의 최소길이를 의미한다.(?)



- QComboBox:editable {background: white;} : editable 은 제일 우측에 있는 화살표 모양의 버튼을 의미하는 것으로 위와 같이 background: white;를 통해 화살표 버튼의 바탕색을 흰색으로 설정하고 있다.



- QComboBox:!editable, QComboBox::drop-down:editable {background: qlineargradient(x1: 0, y1: 0, x2: 0, y2: 1, stop: 0 #E1E1E1, stop: 0.4 #DDDDDD, stop: 0.5 #D8D8D8, stop: 1.0 #D3D3D3);} : 여기서 drop-down:editable 은 콤보박스 배경색의 그라데이션을 의미하는 것으로 콤보박스를 클릭하게 되면 qlineargradient 의 함수에 의해 그라데이션 효과가 나타난다. 그라데이션의 방향은 (x1: 0, y1: 0, x2: 0, y2: 1)에 의해 수직을 의미하며 stop: 0 은 수직으로 내려갈 때의 위치를 비율로 표현한 것이고 그 오른쪽 옆에 있는 16 진수는 색을 의미한다.

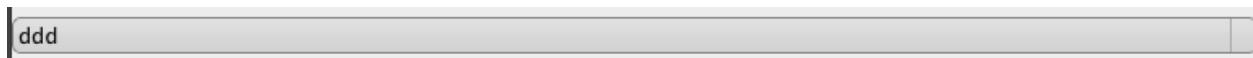


- QComboBox:!editable:on, QComboBox::drop-down:editable:on {background: qlineargradient(x1: 0, y1: 0, x2: 0, y2: 1, stop: 0 #E1E1E1, stop: 0.4 #DDDDDD, stop: 0.5 #D8D8D8, stop: 1.0 #D3D3D3);} : 여기서 맨 첫째 줄에 :on 을 붙이게 되면 콤보박스가 아닌 팝업의 배경색에 그라데이션 효과가 나타난다. 따라서 마찬가지로 qlineargradientdml 함수에 의해 그라데이션 효과가 나타나며 이것 역시 방향은 수직을 의미하고 16 진수를 통해서 색을 입힌다.



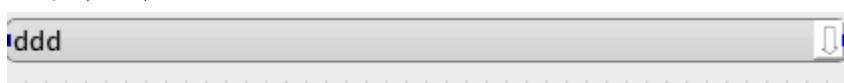
- QComboBox::on {padding-top: 3px; padding-left: 4px;} : 공백 padding 은 글자나 문자의 외곽에 빙 공간을 설정하는 것으로, padding-top 은 위쪽 빙 공간, padding-left 는 아래쪽 공간을 의미한다.

- QComboBox::drop-down {subcontrol-origin: padding; subcontrol-position: top right; width: 15px; border-left-width: 1px; border-left-color: darkgray; border-left-style: solid; border-top-right-radius: 3px; border-bottom-right-radius: 3px;} : 팝업창에서 오른쪽 위의 공백크기를 15px 로 설정하고 border(바깥선)의 왼쪽 가로길이는 1, 색상은 darkgray 이고, 선의 종류는 solid 이다. border(바깥선)의 오른쪽 반원은 맨 첫째 줄에 나오는 3px 로 ComboBox 의 반원과 같게 설정을 해주어야 한다.



- QComboBox::down-arrow {image: url(~~);} : down-arrow 는 콤보박스의 오른쪽에 위치하고 있는 화살표 모양의 클릭버튼이라고 볼 수 있다. 따라서 아래의 그림과 같은 경로의 이미지가 클릭 버튼의 모양이 된다.

- QComboBox::down-arrow:on {top: 1px; left: 1px;} : 팝업창이 열렸을 때 화살표의 모양을 변경시킨다.



04. QDockWidget

DockWidget 은 메인 창 위에 Widget 창을 걸쳐 놓음으로써, 아이콘이나 파일 문서와 같이 편하고 쉬운 관리를 위해서 사용된다.

- QDockWidget {border: 1px solid lightgray; titlebar-close-icon: url(:/new/prefix1/close_button.svg.hi.png); titlebar-normal-icon: url(:/new/prefix1/undock.png);} : 먼저 QDockWidget 클래스 안에서 border 는 바깥선을 뜻하며 1px 의 크기에 solid 라는 선이며, 색상은 밝은 회색이다. 그리고 close 버튼과 normal 버튼의 모양을 위의 경로에 있는 이미지로 바꾸어주고 있다.



- QDockWidget::title {text-align: left; background: lightgray; padding-left: 5px;} :

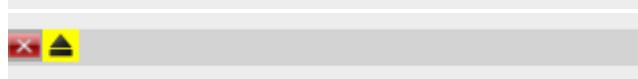
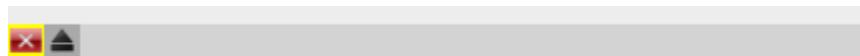
DockWidget 의 title 바를 text-align 에 의해 왼쪽으로 정렬하고 있고, 배경색은 밝은 회색이며, 왼쪽의 여백 공간은 5px 로 설정하고 있다.



- QDockWidget::close-button, QDockWidget::float-button {border: 1px solid transparent; background: darkgray; padding: 0px;} : 맨 위 왼쪽에 있는 close-button 과 float-button 내의 영역 내의 표시를 border(바깥선)과 background: darkgray 를 통해 바깥선은 1px 크기에 solid 선으로 투명하게 표시하였고, 색상은 어두운 회색을 사용하였다. 마지막으로 padding: 0px 를 통해 여백의 공간을 0 으로 설정하였다.



- QDockWidget::close-button:hover, QDockWidget::float-button:hover {background: gray;} : hover 라는 것은 CheckBox 에서 설명했던 것처럼 마우스 포인터를 그 지점에다 갖다 놓으면 색상이나 이미지가 바뀌는 것이라고 하였다. 마찬가지로 마우스 포인터를 close-button 과 float-button 지점에 갖다 놓으면 색깔이 회색으로 변경되는 것을 볼 수 있다.



- QDockWidget::close-button:pressed, QDockWidget::float-button:pressed {padding: 1px -1px -1px 1px;} : 닫기버튼과 float 버튼의 여백공간을 1px 씩 띄우라는 말 같은데 화면에 표시가 나지 않습니다. ㅜㅜ

- QDockWidget::close-button {subcontrol-position: top left; subcontrol-origin: margin; position: absolute; top: 0px; left: 0px; bottom: 0px; width: 14px;} : 닫기 버튼의 위치가 왼쪽 위이고, margin(여백공간)은 위로 0px, 왼쪽으로 0px, 아래로 0px, 가로길이는 14px 이다. 실행화면은 아래와 같다.



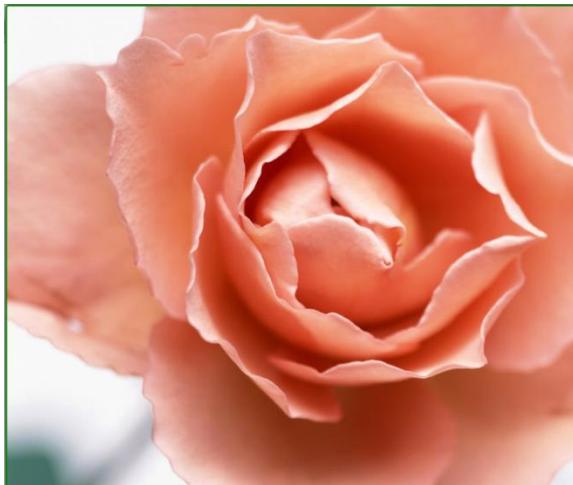
- QDockWidget::float-button {subcontrol-position: top left; subcontrol-origin: margin; position: absolute; top: 0px; left: 16px; bottom: 0px; width: 14px;} : float 버튼의 위치가 왼쪽 위이고, margin(여백공간)은 위로 0px, 왼쪽으로 16px, 아래로 0px, 가로길이는 14px 이다. 실행화면은 아래와 같다.



05. QFrame

Frame 은 어떤 모델링을 하기 전에 기본 틀을 제공해주는 것이다.

- QFrame, QLabel, QToolTip {border: 2px solid green; border-radius: 4px; padding: 2px; background-image: url(:/new/prefix1/background1.jpg);} : border(바깥선)의 크기를 2px 로, border(바깥선) 끝의 반원을 4px 로, 여백 공간을 2px 로 설정해주고 background-image 에서 경로를 통해서 배경이미지를 바꿀 수 있다.



06. QGroupBox

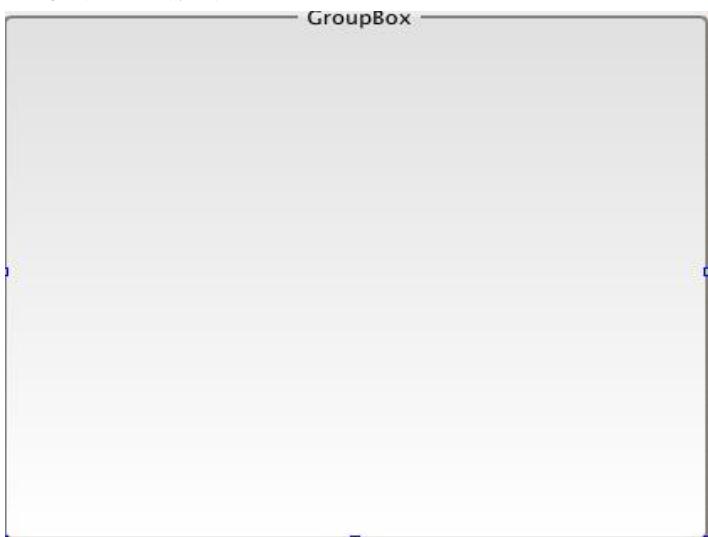
먼저 GroupBox 는 비슷한 기능을 수행하는 컨트롤들을 모아놓은 것이다.

- QGroupBox {background-color: qlineargradient(x1:0, y1:0, x2:0, y2:1, stop: 0 #E0E0E0, stop: 1 #FFFFFF); border: 2px solid gray; border-radius: 5px; margin-top: 1ex;} : 위의 내용은 GroupBox 의 모양을 다음 설명과 같이 설정해주고 있다. 먼저 배경색은 background-color 에 qlineargradient 를 통해 그라데이션 효과를 주고 있다. 그리고 border(바깥선)는 2px 의 크기로, 색상은 회색, 끝선의 반원은 5px, 맨 위의 여유공간을 margin-top 을 통해 설정하고 있다. 여기서 margin-top 을 정확히 설명하자면 GroupBox 와 Layout 의 여유공간을 뜻한다.

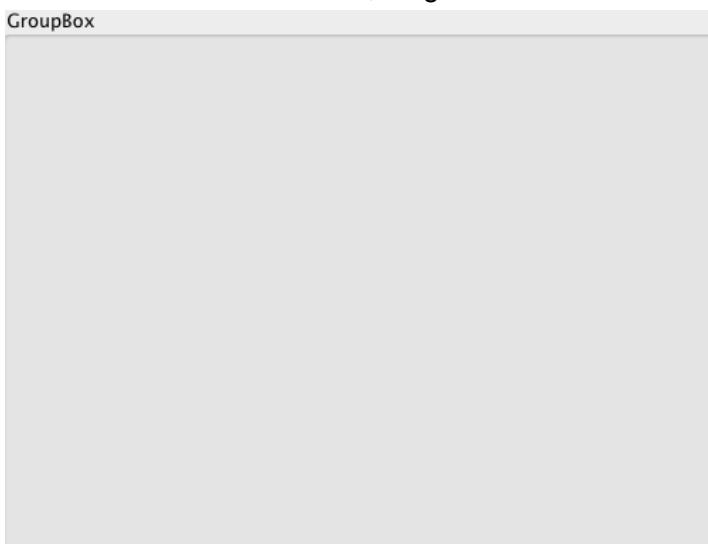


- QGroupBox::title {subcontrol-origin: margin; subcontrol-position: top center; padding: 0 3px;

`background-color: qlineargradient(x1: 0, y1:0, x2:0, y2:1, stop: 0 #FFOECE, stop: 1 #FFFFFF);}` : 다음은 GroupBox 의 맨 위에 있는 타이틀을 설정해주고 있다. 먼저 `subcontrol-origin` 을 통해서 여백의 공간을 설정하고 있다. 그 뒤에 `subcontrol-position` 을 통해서 가운데 정렬로 설정하고, `padding(여백공간)`을 0 3px 로 설정함으로써 'GroupBox'라고 쓰여있는 글자 주변의 `border(바깥선)`을 여백으로 바꾸어주고 있다. 마지막으로 `background-color` 를 통해 타이틀의 배경색을 그라데이션 색상으로 설정해주고 있다.

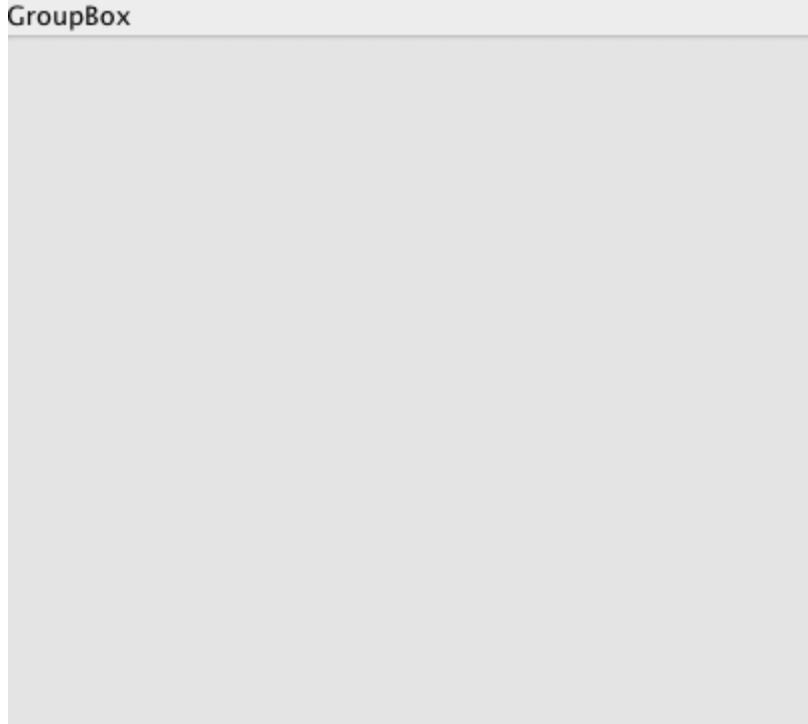


- `QGroupBox::indicator {width: 13px; height: 13px;}` : 'GroupBox' 글자의 위치를 지정해주고 있다. 여기서 `width` 는 가로, `height` 는 세로를 의미한다.



- QGroupBox::indicator:unchecked {image: url(:/new/prefix1/checkbox-unchecked-md.png);} :
unchecked 는 GroupBox 가 체크되지 않았을 때의 상태를 나타내는 것으로서, 위와 같은
경로의 이미지를 보여주게 된다.

GroupBox



07. QHeaderView

HeaderView 란 기준의 TableView에서 행과 열의 첫 번째 컬럼을 뜻하는 즉, 행에서는 “Day, Date, Event”이고 열에서는 ‘1, 2’라 할 수 있겠다.

	Day	Date	Event
1	Monday	25 June	Meeting
2	Tuesday	26 June	Appointment

여기서는 TableView를 Dialog로 구현하지 않고 MainWindow로 구현하였기 때문에 headerview도 마찬가지로 MainWindow로 구현하여 준다. 이러한 heaview를 구현하기 위해서는 사전에 코딩 작업이 필요한데 그 작업 순서는 아래와 같다.

먼저 mainwindows.h에서 class Qheaderview라고 선언을 한 다음 private 영역 안에서 QHeaderView* h라는 객체를 생성하여 준다.

```
1 ifndef MAINWINDOW_H
2 define MAINWINDOW_H
3
4 include < QMainWindow>
5
6 namespace Ui {
7 class MainWindow;
8 }
9 class QHeaderView;
10
11 class MainWindow : public QMainWindow
12 {
13     Q_OBJECT
14
15 public:
16     explicit MainWindow(QWidget *parent = 0);
17     ~MainWindow();
18
19 private:
20     Ui::MainWindow *ui;
21     QHeaderView* h;
22 };
23
24 #endif // MAINWINDOW_H
```

그 후로 mainwindow.cpp에서 #include를 통해서 headerview를 아래 화면과 같이 선언한다.

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include <QTableView>
#include <QStandardItemModel>
#include <QHeaderView>
```

그 다음으로 MainWindow::MainWindow(QWidget *parent) : QMainWindow(parent), ui(new Ui::MainWindow)라는 생성자 함수 안에 아래 화면과 같이 headerview를 선언한다.

```
h = new QHeaderView(Qt::Vertical, parent);
ui->tableView->setVerticalHeader(h);
```

마지막으로 mainwindow.ui로 가서 탑재된 tableView의 styleSheet로 가서 아래의 예제와 같은 내용을 입력하면 디자인을 원하는 대로 설정할 수 있다.

- QHeaderView::section {background-color: qlineargradient(x1: 0, y1: 0, x2: 0, y2: 1, stop: 0 #616161, stop: 0.5 #505050, stop: 0.6#434343, stop: 1 #656565); color: white; padding-left: 4px; border: 1px solid #6c6c6c;} : 행과 열 header 구역별로 디자인을 변경한 것인데 먼저 배경색상은 그라데이션 효과를 입힌 검은색 색상이고, 색상은 흰색, padding-left(왼쪽 여백공간)는 4px, border(바깥선)의 굵기는 1px, 선의 종류는 solid, 색상은 #6c6c6c 이다. 실행 화면은 아래와 같다.

	Day	Date	Event
1	Monday	25 June	Meeting
2	Tuesday	26 June	Appointment

- QHeaderView::section:checked {background-color: red;} : 마우스로 클릭한(선택한) 행의 headerview 배경색상을 빨간색으로 나타내준다. 실행화면은 아래와 같다.

	Day	Date	Event
1	Monday	25 June	Meeting
2	Tuesday	26 June	Appointment

아래의 예제에 있는 down-arrow 와 up-arrow 는 정렬과 관련된 것인데 이 정렬을 사용하기 위해서는mainwindow.ui 의 property 에 있는 Property 의 QtableView에서 sortingEnabled 를 아래의 화면과 같이 체크해주면 된다.

Property	Value
verticalScrollMode	ScrollPerItem
horizontalScrollMode	ScrollPerItem
▼ QTableView	
showGrid	<input checked="" type="checkbox"/>
gridStyle	SolidLine
sortingEnabled	<input checked="" type="checkbox"/>
wordWrap	<input checked="" type="checkbox"/>
cornerButtonEnabled	<input checked="" type="checkbox"/>

- QHeaderView::down-arrow{image: url(:/new/prefix1/arrow-25833_640.png);} : 내림차순 정렬일 때 위의 경로에 있는 이미지 아이콘이 나타난다. 실행화면은 아래와 같다.

	Day	Date	Event
1	Tuesday	26 June	Appointment
2	Monday	25 June	Meeting

- QHeaderView::up-arrow {image: url(:/new/prefix1/up-arrow1.jpeg);} : 오름차순 정렬일 때 위의 경로에 있는 이미지 아이콘이 나타난다. 실행화면은 아래와 같다.

Day	Date	Event
1 Monday	25 June	Meeting
2 Tuesday	26 June	Appointment

08. QLineEdit

LineEdit 은 문자나 숫자를 넣을 수 있는 공간이다.

- QLineEdit {border: 2px solid gray; border-radius: 10px; padding: 0 8px; background: yellow; selection-background-color: darkgray;} : 현재 border(바깥선)의 굵기는 2px, 선의 종류는 solid, 색상은 회색으로 설정하고 있다. 그리고 border(바깥선) 양끝을 10px 의 반지름을 가진 반원으로 설정하였으며 padding 을 통해 여백공간을 설정해주고 있다. 마지막으로 LineEdit 의 배경색은 노란색으로 설정하고 selection-background-color 를 통해 LineEdit 을 감싸고 있는 바깥선을 어두운 회색으로 설정하였다.



- QLineEdit[echoMode = "2"] {lineedit-password-character: 9679;} : 먼저 echoMode 를 사용하기 위해서 dialog.cpp 에서 함수를 설정해 주어야 한다. 함수를 설정하는 방법은 Dialog 클래스 안에 ui->lineedit->setEchoMode(QLineEdit::Password);와 같다. 함수를 설정한 뒤에 위와 같은 내용으로 LineEdit 에 비밀번호를 치면 숫자가 보이지 않고 별표시가 대신 나온다.

```
ui->lineEdit->setEchoMode(QLineEdit::Password);
```



- QLineEdit::read-only {background: lightblue;} : read-only 는 읽기 전용으로써, 쓰기가 불가능하고 읽는 것만 가능하다. 따라서 읽기 전용의 LineEdit 은 배경색이 파란색임을 알 수 있다. 그리고 read-only 를 사용하기 위해서는 dialog.cpp 에서 ui->lineedit->setReadOnly(true);라는 명령을 해주어야 한다.

```
ui->lineEdit->setReadOnly(true);
```



09. QListView

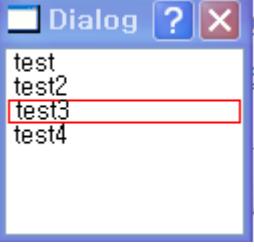
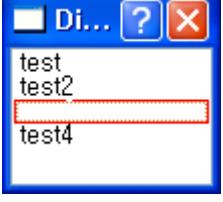
- `QListView { alternate-background-color: yellow; } :`

- `QListView { show-decoration-selected: 1; } :` 리스트에 있는 항목을 선택했을 시에 그 항목만 영역 선택되게 할 것인지, 아니면 한 줄의 길이만큼을 영역 선택 할 것인지 정하는 것 같다.

	
<code>show-decoration-selected: 1;</code>	<code>show-decoration-selected: 0;</code>

- `QListView::item:alternate { background: #EEEEEE; } :` 색상을 변경해도 변화가 없어서 확인할 수 없습니다.

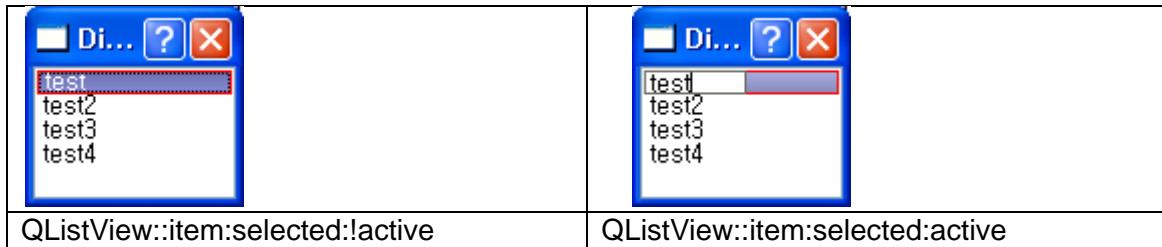
- `QListView::item:selected { border:1px solid red; } :` 리스트에 있는 항목을 선택했을 때에 대한 설정입니다. 위와 같은 경우에는 선택했을 때 1px 굵기의 빨간 테두리 선이 생성됩니다.

	
Selected 적용 결과	활성화 창일 경우 선택할 글자가 안보임

- `QListView::item:selected:!active{background: qlineargradient(x1 : 0, y1 : 0, x2 : 0, y2 : 1, Stop : 0 #ABAFAE5, stop: 1 #8588B2 } }`

- `QListView::item:selected:active{background: qlineargradient(x1 : 0, y1 : 0, x2 : 0, y2 : 1, Stop : 0 #6a6ea9, stop: 1 #888dd9 } }`

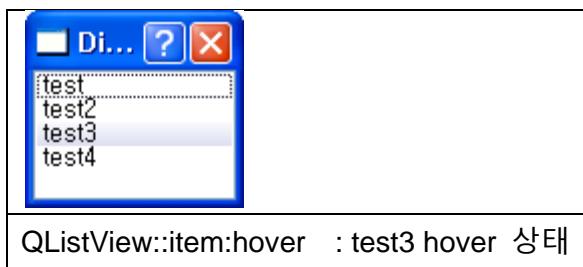
항목이 선택되었을 때 수정 중일 때를 active, 그냥 선택만 되어 있을 때는 !active로 그 상태에 따른 디자인을 설정할 수 있습니다. 위에 select 의 문제점이 사라진 것을 확인할 수 있습니다. 그리고 색상의 경우에 qlineargradient 함수를 이용해서 그라데이션이 입혀집니다. x1, x2 는 0 으로 동일하며 y1, y2 는 0 과 1 이므로 수직으로 입혀지며 그 옆의 의미는 색상입니다.



QListView::item:selected:!active

- QListView::item:hover{background: qlineargradient(x1 : 0, y1 : 0, x2 : 0, y2 : 1,

Stop : 0 #FAFBFE, stop: 1 #DCDEF1 } : hover 의 경우에는 마우스가 올려져 있을 때의 상태이다. 항목 위에 마우스가 올려져 있다면 위와 같은 경우에는 qlineargradient 함수를 이용해서 그라디에이션이 수직으로 입혀지게 된다.



QListView::item:hover : test3 hover 상태

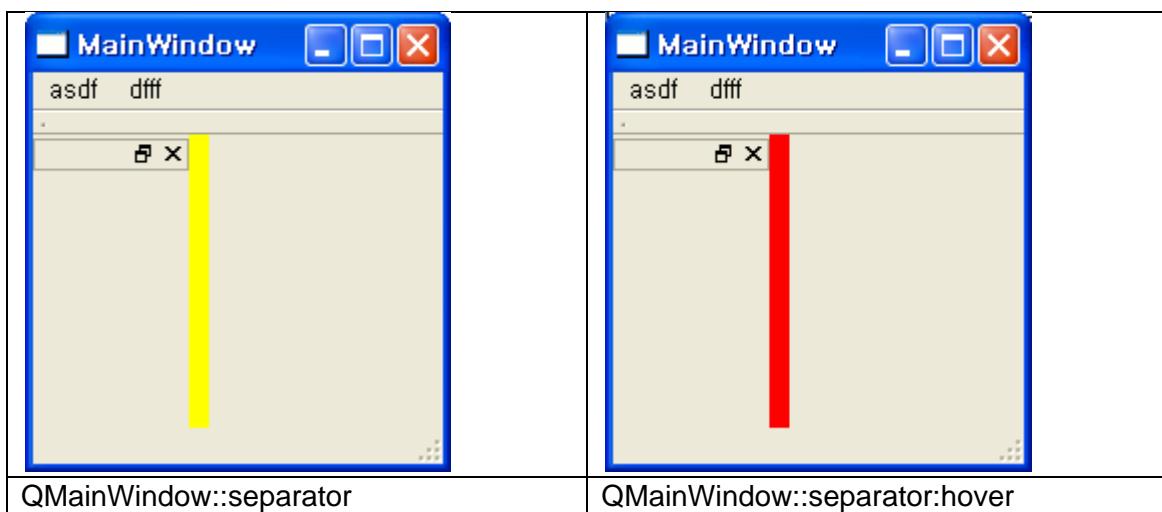
10. QMainWindow

- QMainWindow::separator{ background:yellow; width:10px; height:10px; }

MainWindow 에 Dock Widget 이 올라온 경우에 Separator 를 이용해서 그 경계에 대한 디자인을 설정할 수 있다. 위의 경우에는 경계선이 노란색이 되며 굵기가 10px 가 된다.

- QMainWindow::separator:hover{ background:red; }

위와 같이 경계선을 만든 상황에서 경계선에 마우스가 올라간 hover 상태일 때 디자인이다. 예시에서는 빨간색으로 변하게 설정했다.

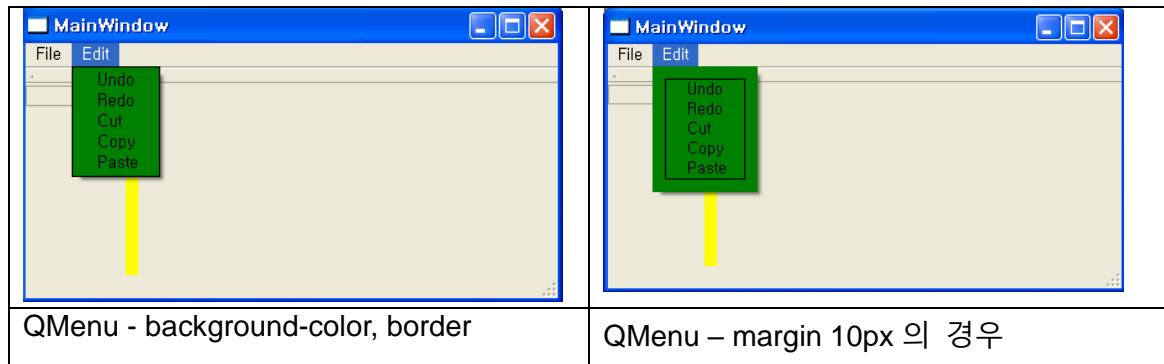


11. QMenu

- QMenu { background-color:green; border:1px solid black; }

MainWindow에서 위쪽에 메뉴에 대한 설정을 할 수 있다. Background-color 설정을 하게 되면 메뉴 내에 색상이 바뀌게 된다. 그리고 테두리 선은 1px 검은 선이다.

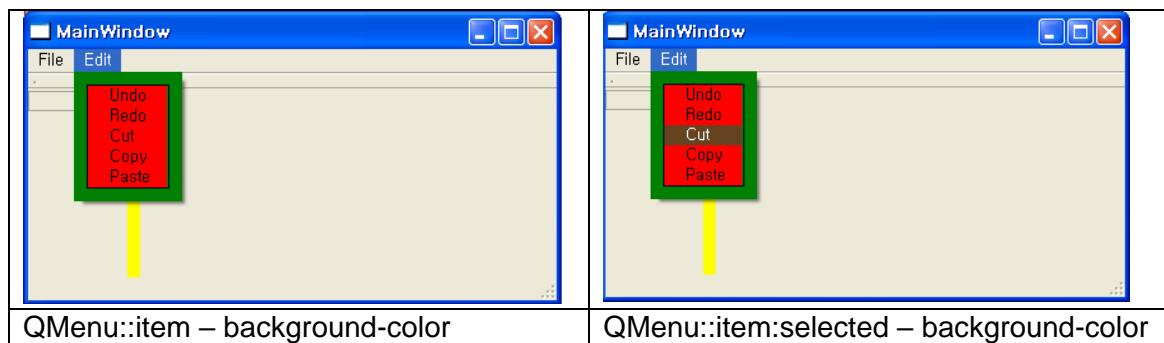
- QMenu { margin 10px; } : 메뉴 창 안에서 상하좌우 여백을 두기 위한 것이다 아래 그림에선 알아보기 쉽게 하기 위해 10px로 설정해서 알아봤다.



- QMenu::item {background-color:red;}

- QMenu::item:selected { background-color:#654321; }

QMenu의 안에 항목들을 디자인 설정할 수 있는 것이 QMenu::item이다. 여기서는 background-color를 red로 만들어 빨간색으로 나타냈다. 또한 QMenu::item:selected로 항목이 선택되었을 때 디자인을 설정할 수 있다. 여기서는 그 항목의 배경 색을 #654321로 수정해서 나타내봤다.



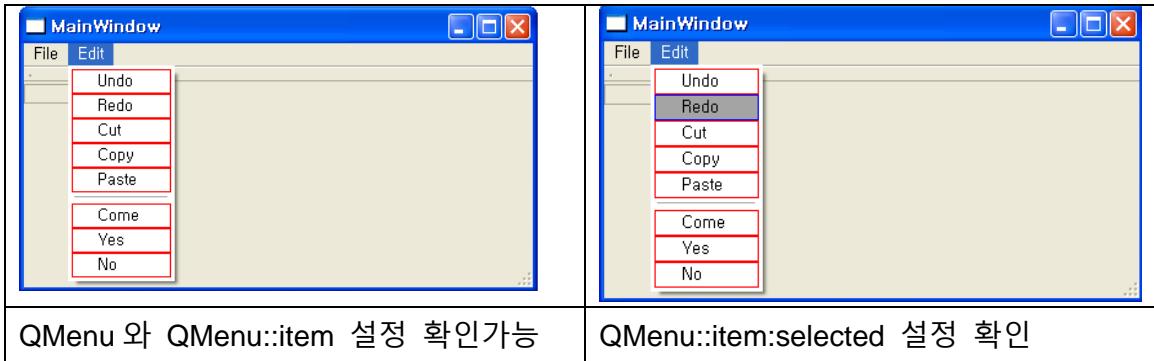
- QMenu { background-color:white; margin 3px; }

- QMenu::item { padding: 3px 25px 3px 20px ; border:1px solid red; }

- QMenu::item:selected { border-color:blue; background:rgba(100,100,100,150); }

QMenu의 배경색은 white로 하고 margin은 3px로 간격을 설정했다. 그리고 QMenu::item에서 padding 설정이 나오는데 이 설정은 각 메뉴 안의 메뉴이름 각각 하나하나에 top left bottom right 부분의 간격을 설정한 것이다. 그리고 테두리선은 빨간색 굵기는 1px로 설정했다.

QMenu::item:selected 에서는 테두리 선의 색상을 파란색으로 변경하며 배경색을 rgba 색상표에 적힌 값에 맞게 변하게 설정했다.



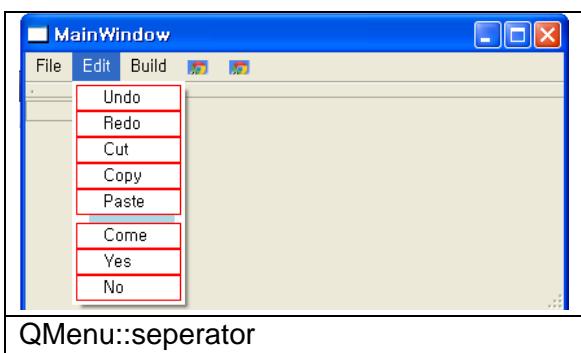
- QMenu::icon:checked { background:gray; border: 1px inset gray; position: absolute; top:1px, right: 1px; bottom: 1px, left: 1px}

QAction	
checkable	<input checked="" type="checkbox"/>
checked	<input type="checkbox"/>
enabled	<input checked="" type="checkbox"/>
icon	radiobutton_un...

메뉴바 아래의 메뉴에 Icon 과 Check 속성을 사용하고 그에 대한 디자인을 할 때 필요한 QSS 구문이다. Checkable 을 체크하고 checked 를 체크하면 체크가 된 상태로 된다. Icon 은 원하는 이미지로 선택하면 그 이미지로 선택된다.

여기서 QSS 구문 icon:checked 는 check 된 상태일 때의 디자인을 의미한다. 배경색을 회색으로 하고 테두리 선을 1px 굵기에 inset 방식(내용이 들어가게 보임) 색상은 회색으로 보이게 한다. 그리고 절대좌표로 top , right, bottom, left 로 1px 의 여백을 둔다.

- QMenu::separator { height:6px; background: lightblue; margin-left:10px; margin-right:5px; }
Seperator 는 앞서 MainWindow 에서도 나왔듯이 구분선을 지칭하는 것이다. 그래서 QMenu 에 있는 Seperator 는 메뉴와 메뉴 사이에 구분선에 대한 디자인을 정의할 수 있다. 여기서 height 는 굵기, background 는 색상, margin-left, margin-right 는 적힌 px 간격만큼 좌우로 여백을 두는 것이다.



- QMenu::indicator {width: 13px; height: 13px;}

메뉴의 Check 부분을 활성화하면 그 Check 박스의 크기를 설정한다.

```
- QMenu::indicator:non-exclusive:unchecked {image:url(:/new/prefix1/checkbox_unchecked (1).png);}
- QMenu::indicator:non-exclusive:unchecked:selected{image:url(:/new/prefix1/checkbox_unchecked_hover.png);}
- QMenu::indicator:non-exclusive:unchecked:pressed {image: url(:/new/prefix1/checkbox_unchecked_pressed.png);}
- QMenu::indicator:non-exclusive:checked {image: url(:/new/prefix1/checkbox_checked (1).png);}
- QMenu::indicator:non-exclusive:checked:selected {image: url(:/new/prefix1/checkbox_checked_hover.png);}
- QMenu::indicator:non-exclusive:checked:pressed {image: url(:/new/prefix1/checkbox_checked_pressed.png);}
- QMenu::indicator:exclusive:unchecked {image: url(:/new/prefix1/radiobutton_unchecked.png);}
- QMenu::indicator:exclusive:unchecked:selected {image: url(:/new/prefix1/radiobutton_unchecked_hover.png);}
- QMenu::indicator:exclusive:unchecked:pressed {image: url(:/new/prefix1/radiobutton_unchecked_pressed.png);}
- QMenu::indicator:exclusive:checked {image: url(:/new/prefix1/radiobutton_checked.png);}
- QMenu::indicator:exclusive:checked:selected {image: url(:/new/prefix1/radiobutton_checked_hover.png);}
- QMenu::indicator:exclusive:checked:pressed {image: url(:/new/prefix1/radiobutton_checked_pressed.png)}
```

메뉴 Exclusive 기능은 메뉴를 그룹화 시켜야 사용할 수 있는 것이다. QActionGroup 으로 메뉴들을 묶어준 다음 각각의 상태마다 이미지를 넣어준다.

Non-exclusive: 다중 체크박스 (QActionGroup 설정-> setExclusive(false)로 설정

Exclusive : 그룹화 된 것 중 하나만 선택 되는 체크박스 (setExclusive(true))

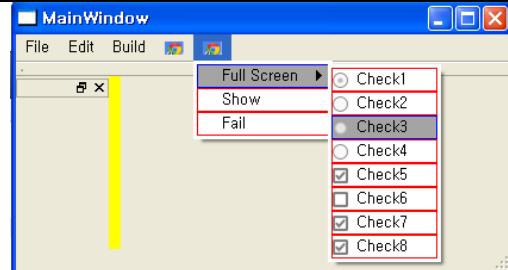
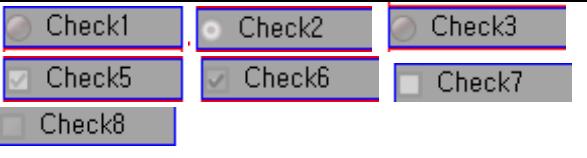
unchecked : 비선택 , unchecked:selected : 빈 체크박스에 마우스가 올라온 경우

unchecked:pressed : 비선택시에 마우스를 클릭한경우, checked : 선택

checked:selected : 선택 된 체크박스에 마우스가 올라온 경우

checked:pressed : 선택 된 체크박스에 마우스를 클릭한 경우

Exclusive 의 경우 하나만 선택하는 라디오 버튼과 비슷한 기능이라 위 코드에서와 같이 라디오 버튼 이미지를 가져와서 사용했습니다.

	
Check1 : Exclusive:checked Check2, 4 : Exclusive:unchecked Check3 : Exclusive:unchecked:selected Check5, 7, 8 : Non-exclusive:checked Check6 : Non-exclusive:unchecked	Check1 : Exclusive:checked:pressed Check2 : Exclusive:checked:selected Check3 : Exclusive:unchecked:pressed Check5 : Non-Exclusive:checked:selected Check6 : Non-Exclusive:checked:pressed Check7 : Non-Exclusive:unchecked:selected Check8 : Non-Exclusive:unchecked:pressed

12. QMenuBar

- QMenuBar { background-color:qlineargradient(x1:0, y1:0, x2:0, y2:1, stop 0 lightgray, stop 1 darkgray); }

메뉴바의 디자인 설정입니다. 위와 같이 입력하게 되면 프로그램 메뉴바의 색상이 qlineargradient 함수에 의해 수직방향으로 lightgray -> darkgray로 그라데이션 되게 됩니다.

- QMenuBar::item{ spacing: 3px; padding 1px 4px; background:transparent; border-radius:4px; }

메뉴바의 각 속성인 메뉴에 대한 디자인 설정인 것 같습니다.

Spacing : 메뉴와 메뉴 사이의 간격을 조정하는 것으로 3px 간격을 두었다.

Padding 1px 4px : top bottom 간격을 1px로 설정, left right 간격을 4px로 설정

Background : 배경을 투명(transparent)하게

Border-radius : 메뉴의 양 끝을 4px 길이의 반지름을 가진 원으로 표현 (둥글게)

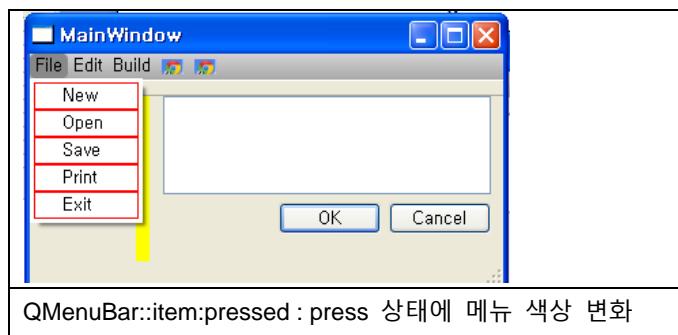
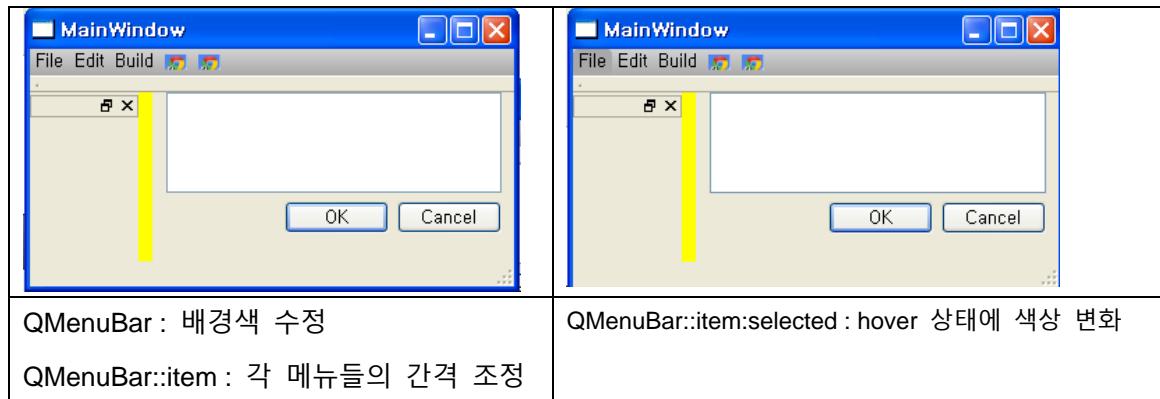
- QMenuBar::item:selected{ background:#a8a8a8; }

- QMenuBar::item:pressed{ background:#888888; }

메뉴 바에 있는 속성(메뉴)을 hover, press 상태일 경우에 디자인을 설정하는 방법이다.

Selected(hover): 배경 색상을 #a8a8a8로 변환

Pressed(press): 배경 색상을 #888888로 변환



13. QProgressBar

ProgressBar 는 어떤 양의 비율(퍼센티지)을 보여주는 하나의 도구이다.

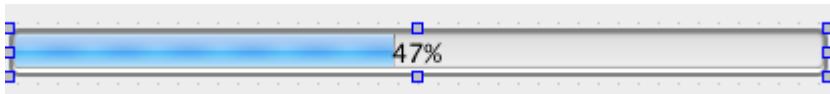
- **QProgressBar {border: 2px solid grey; border-radius: 5px;}** : 현재 border(바깥선)의 굵기는 2px, 색상은 회색, 양쪽 끝의 반원 반지름을 5px로 설정하고 있다.



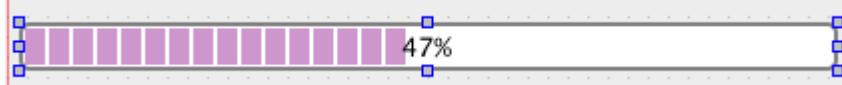
- **QProgressBar::chunk {background-color: #05B8CC; width: 20px;}** : ProgressBar에서 Chunk는 가로표시줄을 채우는 어떤 비율을 보여주는 도구이다. 따라서 background-color를 통해 Chunk의 색상을 청록색으로 바꾸어 주었고, 가로 길이는 20px로 설정하였다.



- **QProgressBar {border: 2px solid grey; border-radius: 5px; text-align: center;}** : border(바깥선)의 굵기는 2px, 색상은 회색, 양쪽 끝의 반원 반지름을 5px로 설정하고 있다. 마지막으로 '47%'라고 표시되어 있는 문자를 text-align을 이용해 가운데 정렬하고 있다.



- **QProgressBar::chunk {background-color: #CD96CD; width: 10px; margin: 0.5px;}** : 현재 chunk(가로표시줄)을 설정하고 있는데 색상은 분홍색, 가로길이는 10px, 가로표시줄의 여백은 0.5px로 설정하고 있다.



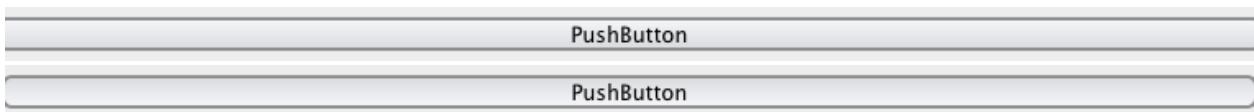
14. QPushButton

PushButton 은 보통 흔히 일상에서 보는 버튼을 말하는 것으로, 버튼의 모양을 설정할 수 있다.

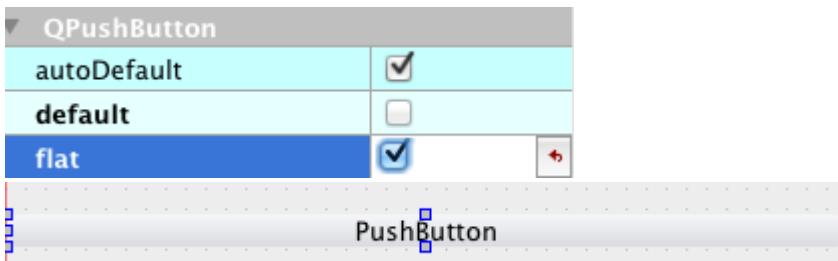
- QPushButton {border: 2px solid #8f8f91; border-radius: 6px; background-color: qlineargradient(x1: 0, y1: 0, x2:0, y2:1, stop: 0 #f6f7fa, stop: 1 #dadbdbe); min-width: 80px;} : 먼저 버튼의 border(바깥선)의 굵기를 2px 로, 색상은 회색, 양쪽 끝의 반원 반지름을 6px 로, 버튼의 배경색을 qlineargradient 를 통해서 그라데이션 효과를 주고 있다. 마지막으로 버튼의 가로 최소길이를 80px 로 설정하고 있다.



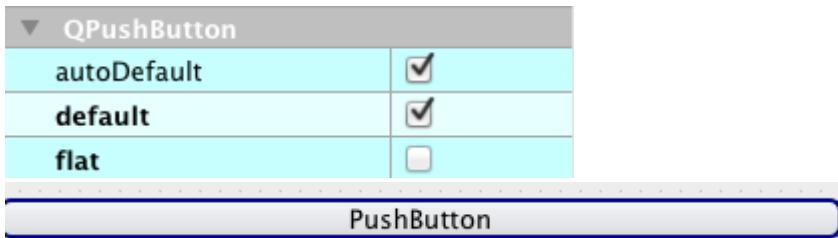
- QPushButton::pressed {background-color: qlineargradient(x1: 0, y1: 0, x2: 0, y2: 1, stop: 0 #dadbdbe, stop: 1 #f6f7fa);} : 버튼을 눌렀을 때의 색상효과를 나타낸 것으로 qlineargradient 를 통해 그라데이션 효과를 주고 있으며, 버튼을 눌렀을 때와 누르지 않았을 때의 차이를 색상을 통해서 보여주고 있다. (눌렀을 때와 누르지 않았을 때의 색상차이가 보인다.)



- QPushButton::flat {border: none;} : Qt Programming 에서 만약에 Property Editor 를 flat 으로 설정해 준다면 위와 같이 border(바깥선)은 none 이므로 border(바깥선)가 없어진다.



- QPushButton::default {border-color: navy;} : Property Editor 를 default 로 설정하였다면 border(바깥선)의 색상은 남색이 된다.



*pushButton 이미지 넣기

- QPushButton {color: black; border-image: url(:/new/prefix1/button1.jpeg) 3 10 3 10; border-top: 3px transparent; border-bottom: 3px transparent; border-right: 10px transparent; border-left: 10px transparent;} : pushButton 에다 위와 같은 경로의 이미지를 넣고 border(바깥선) 위는 3px 만큼

투명하게, border(바깥선) 아래는 3px 만큼 투명하게, border(바깥선) 오른쪽은 10px 만큼 투명하게, border(바깥선) 왼쪽은 10px 만큼 투명하게 만들어준다. 실행화면은 아래와 같다.



15. QRadioButton

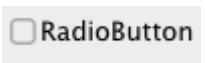
RadioButton 은 윈도우나 디어얼로그의 박스 안에서 어느 하나를 선택, 취소하기 위한 버튼이다.

- **QRadioButton::indicator {width: 13px; height: 13px;}** : 따라서 일련의 선택사항 중 하나만 선택할 수 있다. 이런 Radio 버튼의 크기를 가로 13px, 세로 13px 로 설정하였다.

- **QRadioButton::indicator::unchecked {image: url(:/new/prefix1/checkbox-unchecked-md.png);}** : RadioButton 을 체크하지 않았을 때 위와 같은 경로안에 있는 파일의 이미지가 뜨게 된다.



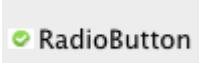
- **QRadioButton::indicator:unchecked:hover {image: url(:/new/prefix1/radiobutton_unchecked.svg);}** : RadioButton 을 체크하지 않은 상태에서 마우스 포인터를 RadioButton 에다 놓아두면 아래의 그림과 같은 경로의 파일 이미지가 뜨게 된다.



- **QRadioButton::indicator:unchecked:pressed {image: url(:/new/prefix1/unchecked_press.svg);}** : RadioButton 을 체크하기 위해 클릭을 하는 동안 아래의 그림과 같은 경로의 파일 이미지가 뜨게 된다.



- **QRadioButton::indicator::checked {image: url(:/new/prefix1/checked.jpeg);}** : RadioButton 을 체크했을 때 아래의 그림과 같은 경로의 파일 이미지가 뜨게 된다.



- **QRadioButton::indicator:checked:hover {image: url(:/new/prefix1/checked_hover.png);}** : RadioButton 을 클릭하고 마우스 포인터를 RadioButton 에다 놓아두었을 때 아래의 그림과 같은 경로의 파일 이미지가 뜨게 된다.



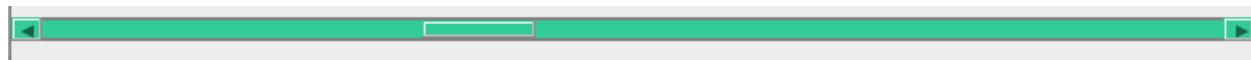
- QRadioButton::indicator:checked:pressed {image: url(:/new/prefix1/checked_press.jpeg);} : RadioButton 이 체크된 상태에서 마우스 클릭을 하는 동안 아래의 그림과 같은 경로의 파일 이미지가 뜨게 된다.



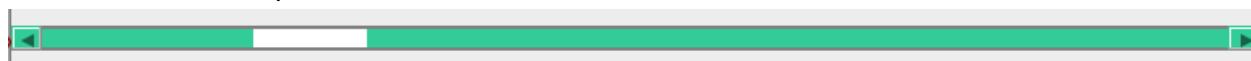
16 - 1. QScrollBar(Horizontal)

스크롤바는 그림이나 문자를 출력할 경우 운영체제 나타난 화면을 상하 또는 좌우로 움직일 때 사용하는 막대이다.

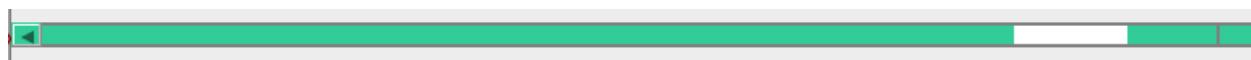
- QScrollBar::horizontal {border: 2px solid grey; background: #32CC99; height: 15px; margin: 0px 20px 0 20px;} : 현재 막대는 horizontal(수평막대)이며 border(바깥선)의 굵기는 2px, 선의 종류는 회색, 배경색은 초록색이며, 세로의 길이가 15px, margin(여백공간)은 가로 0px, 세로 20px 이다.



- QScrollBar::handle:horizontal {background: white; min-width: 20px;} : 스크롤바에서 좌우로 움직일 수 있는 또 하나의 막대(handle)를 만든 것이다. 이 막대의 배경색은 흰색, 가로 최소길이는 20px 이다.

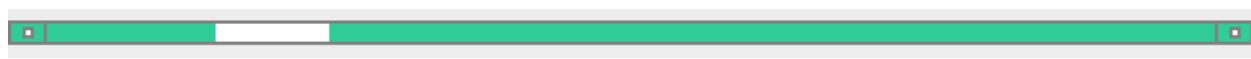


- QScrollbar::add-line:horizontal {border: 2px solid grey; background: #32CC99; width: 20px; subcontrol-position: right; subcontrol-origin: margin;} : 화살표 버튼의 border(바깥선)를 굵기는 2px, 선의 종류는 solid, 배경색상은 초록색, 가로길이는 20px로 설정해 준다. 그리고 subcontrol-position 을 오른쪽으로 설정함으로써 오른쪽 끝에 있는 화살표를 가리키게 된다.

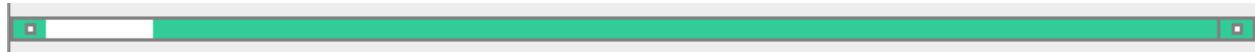


- QScrollBar::sub-line:horizontal {border: 2px solid grey; background: #32CC99; width: 20px; subcontrol-position: left; subcontrol-origin: margin;} : 화살표 버튼의 border(바깥선)를 굵기는 2px, 선의 종류는 solid, 배경색상은 초록색, 가로길이는 20px로 설정해준다. 그리고 subcontrol-position 을 왼쪽으로 설정함으로써 왼쪽 끝에 있는 화살표를 가리키게 된다.

- QScrollBar:left-arrow:horizontal, QScrollBar::right-arrow:horizontal {border: 2px solid grey; width: 3px; height: 3px; background: white;} : 왼쪽과 오른쪽의 화살표를 직접 설정해서 만든 것으로 왼쪽과 오른쪽 화살표의 모양은 가로(width)와 세로(height)의 길이가 같은 정사각형이고, 이 정사각형의 border(바깥선) 굵기는 2px, 선의 종류는 solid, 색상은 회색이며, 마지막으로 이 정사각형의 배경색상은 흰색이다.



- QScrollBar::add-page:horizontal, QScrollBar::sub-page:horizontal {background: none;} : 가운데 있는 표시바(add-page, sub-page)의 색상을 투명색이므로 기준의 초록색이 계속 나타나고 있음을 알 수 있다.



다음은 맥 OS에서 쓰이는 스크롤바 막대를 만들어 보겠다.

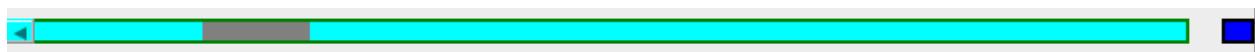
- QScrollBar::horizontal {border: 2px solid green; background: cyan; height: 15px; margin: 0px 40px 0 0px;} : 스크롤바의 border(바깥선)의 굵기는 2px, 색상은 초록색, 선의 종류는 solid이며, 세로의 길이는 15px이며, margin(여백공간)의 가로길이는 0px, 세로길이는 40px로 설정되고 있다. 마지막으로 배경화면 색은 청록색으로 설정하고 있다.



- QScrollBar::handle:horizontal {background: gray; min-width: 20px;} : 스크롤바를 좌우로 움직일 수 있는 막대로써, 이 막대의 배경색상은 회색, 가로의 최소길이는 20px로 설정한 것이다.



- QScrollBar::add-line:horizontal {border: 2px solid black; background: blue; width: 16px; subcontrol-position: right; subcontrol-origin: margin;} : 화살표 버튼의 border(바깥선)를 굵기는 2px, 선의 종류는 solid, 배경색상은 검정색, 가로길이는 16px로 설정해 준다. 그리고 subcontrol-position 을 오른쪽으로 설정함으로써 오른쪽 끝에 있는 화살표를 가리키게 된다.



- QScrollBar::sub-line:horizontal {border: 2px solid black; background: magenta; width: 16px; subcontrol-position: top right; subcontrol-origin: margin; position: absolute; right: 20px;} : 화살표 버튼의 border(바깥선)를 굵기는 2px, 선의 종류는 solid, 배경색상은 Mazenta, 가로길이는 20px로 설정해준다. 그리고 subcontrol-position 을 오른쪽 top 으로 설정함으로써 오른쪽 top 에 있는 화살표를 가리키게 된다.



- QScrollBar:left-arrow:horizontal, QScrollBar::right-arrow:horizontal {width: 3px; height: 3px; background: pink;} : 왼쪽과 오른쪽의 화살표를 직접 설정해서 만든 것으로 왼쪽과 오른쪽 화살표의 모양은 가로(width)와 세로(height)의 길이(3px)가 같은 정사각형이고, 이 정사각형의 배경색상은 흰색이다.



- QScrollBar::add-page:horizontal, QScrollBar::sub-page:horizontal {background: none;} : 가운데 있는 표시바(add-page, sub-page)의 색상이 투명색이므로 기준의 청록색이 계속 나타나고 있음을 알 수 있다.



16 - 2. QScrollBar(Vertical)

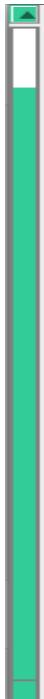
- **QScrollBar::vertical {border: 2px solid grey; background: #32CC99; width: 15px; margin: 22px 0 22px 0;}** : horizontal 이 수평이었다면 vertical 은 수직을 의미한다. 이러한 수직막대의 border(바깥선) 굵기를 2px, 선의 종류는 solid, 배경화면은 초록색으로 설정하였다. 그리고 가로의 길이는 15px 이고 margin(여백공간)은 가로 22px, 세로 0px 이다.



- **QScrollBar::handle:vertical {background: white; min-height: 20px;}** : 스크롤바를 상하로 움직일 수 있는 막대로써, 이 막대의 배경색상은 흰색, 세로 최소길이는 20px 이다.



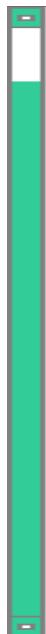
- QScrollBar::add-line:vertical {border: 2px solid grey; background: #32CC99; height: 20px; subcontrol-position: bottom; subcontrol-origin: margin;} : 화살표 버튼의 border(바깥선)를 굵기는 2px, 선의 종류는 solid, 배경 색상은 초록색, 세로길이는 20px로 설정해 준다. 그리고 subcontrol-position 을 아래쪽으로 설정함으로써 아래쪽 끝에 있는 화살표를 가리키게 된다.



- QScrollBar::sub-line:vertical {border: 2px solid grey; background: #32CC99; height: 20px; subcontrol-position: top; subcontrol-origin: margin;} : 화살표 버튼의 border(바깥선)를 굵기는 2px, 선의 종류는 solid, 배경 색상은 초록색, 세로길이는 20px로 설정해 준다. 그리고 subcontrol-position 을 위쪽으로 설정함으로써 위쪽 끝에 있는 화살표를 가리키게 된다.



- QScrollBar::up-arrow:vertical, QScrollBar::down-arrow:vertical {border: 2px solid grey; width: 3px; height: 3px; background: white;} : 위쪽과 아래쪽의 화살표를 직접 설정해서 만든 것으로 위쪽과 아래쪽 화살표의 모양은 가로(width)와 세로(height)의 길이(3px)가 같은 정사각형이고, border(바깥선)의 굵기는 2px, 선의 종류는 solid, 색상은 회색이다. 마지막으로 이 정사각형의 배경색상은 흰색이다.



- QScrollBar::add-page:vertical, QScrollBar::sub-page:vertical {background: none;} : 가운데 있는 표시바(add-page, sub-page)의 색상이 투명색이므로 기존의 초록색이 계속 나타나고 있음을 알 수 있다.



17. QSizeGrip

SizeGrip은 Window 창 크기 조절하는 영역을 설정하는 것이다.

```
QSizeGrip* g;
```

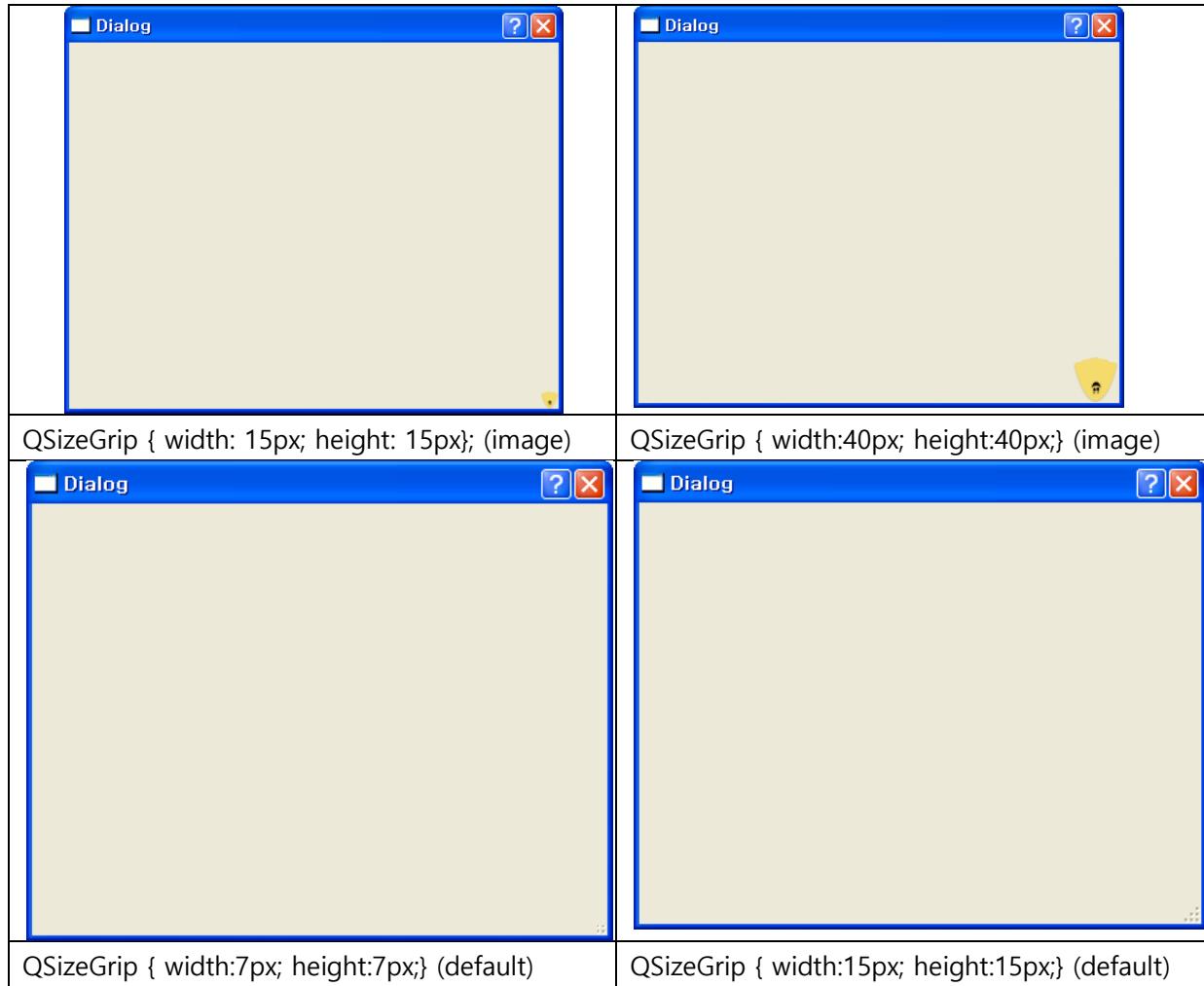
header에서 QSizeGrip을 설정해준다.

```
g = new QSizeGrip(this);  
ui->verticalLayout->addWidget(g, 0, Qt::AlignBottom | Qt::AlignRight);
```

cpp에서 SizeGrip을 만들어주고 우측 하단에 추가해준다.

-QSizeGrip { image: url(:/new/prefix1/spindown.png); width: 40px; height 40px; }

SizeGrip 영역을 QSS로 디자인 할 수 있다. 여기서는 SizeGrip의 기본적인 표시를 그림으로 변경하고 그 영역의 크기를 width와 height로 표현했다.



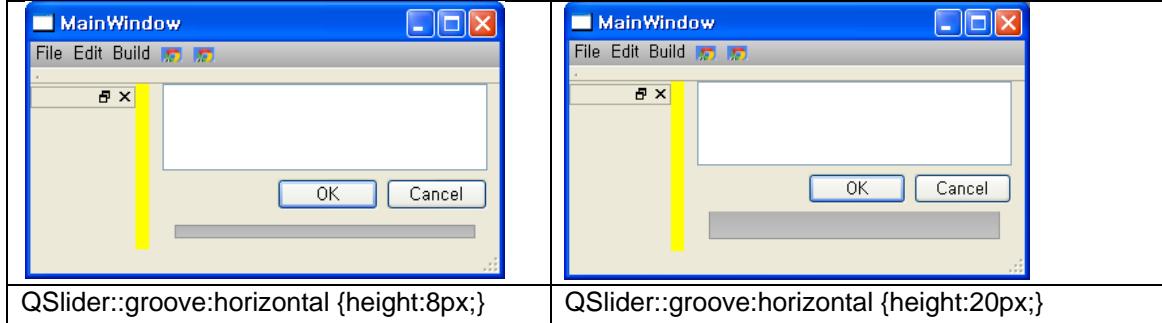
18. QSlider

- `QSlider::groove:horizontal {border:1px solid #999999; height:8px; background:qlineargradient(x1:0, y1:0, x2:0, y2:1, stop:0 #B1B1B1, stop:1 #C4C4C4); margin 9px; }`

QSlider의 겉 테두리 상자부분을 디자인하기 위해서는 QSlider::groove를 이용해야 한다. 여기서 horizontal은 수평(가로) 슬라이더고 vertical은 수직(세로) 슬라이더를 의미한다.

Border의 경우 겉 상자의 테두리 선을 설정하기 위함이고 1px 굵기에 #999999색상을 사용했다.

Height는 슬라이더의 높이를 설정하는 것이다.



Background는 슬라이더의 핸들러 부분을 제외한 나머지 부분의 색상을 선택하는 것이다. 여기서는 qlineargradient 함수를 이용해서 수직방향으로 #B1B1B1 -> #C4C4C4로 그라데이션 했다.

Margin의 경우에는 0으로 둘 경우 정해진 레이아웃에 가득 차게 보이게 되며 간격을 두면 양 쪽에 여백을 두기 때문에 디자인 할 때 보기 좋게 할 수 있다.

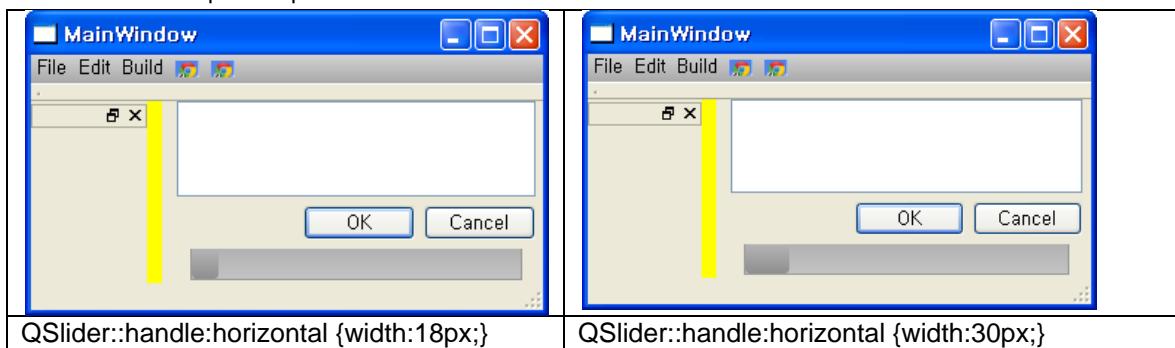
여기서 핸들러 부분이 나오지 않게 되는데 따로 핸들러에 대한 코드를 추가해줘야 된다. (배경에 덮여서 보이지 않는다.)

- `QSlider::handle:horizontal { background:qlineargradient(x1:0, y1:0, x2:0, y2:1, stop:0 #B4B4B4, stop:1 #8f8f8f); border 1px solid #5c5c5c; width:18px; border-radius:3px; }`

Background는 핸들러의 색상을 지정하는 부분이다. Qlineargradient 함수를 이용해서 수직방향으로 #b4b4b4 -> #8f8f8f로 그라데이션 했다.

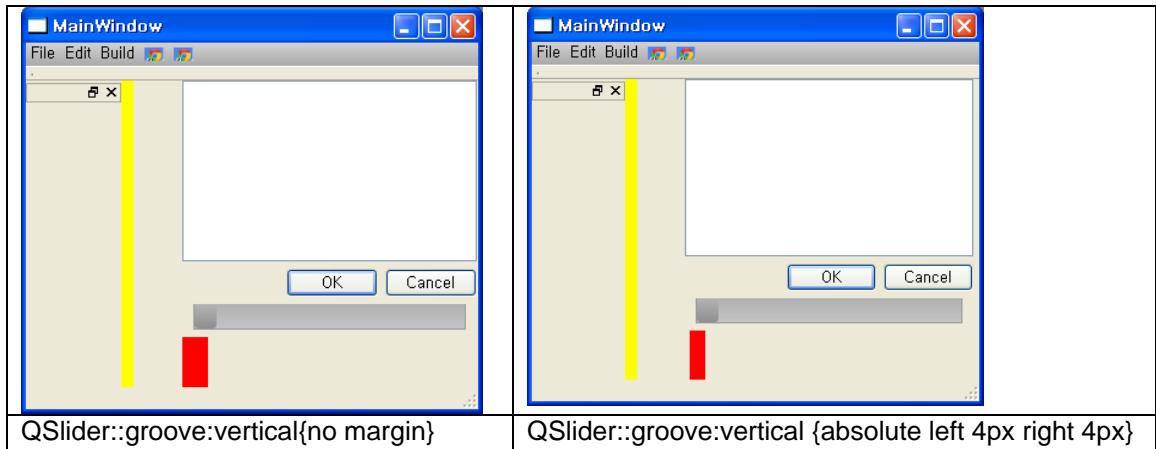
테두리의 경우 #5c5c5c 색상으로 1px 굵기이며, 핸들러의 너비는 18px로 설정되었다.

Border-radius 3px로 3px를 반지름으로 하는 원으로 핸들러의 양끝을 둥글게 표현했다.



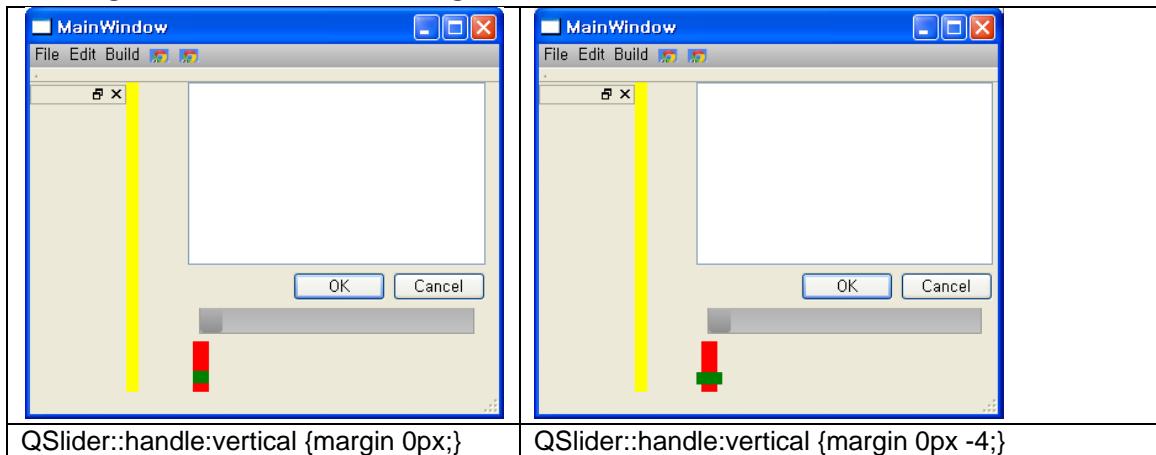
- `QSlider::groove:vertical { background:red; position:absolute; left:4px; right:4px; }`

이번엔 Vertical(수직) 슬라이더를 연습해본다. groove부분의 색상을 red로 하고 좌, 우 여백을 절대좌표로 4px씩 둔다. 이렇게 두게 되면 핸들러가 groove 부분보다 길어져서 익숙하게 써왔던 슬라이더가 된다.



- QSlider::handle:vertical { height: 10px; background:green; margin: 0px -4;}

이번엔 앞서 만든 vertical 슬라이더에 핸들러 디자인을 추가한다. 높이는 10px, 색상은 녹색, 그리고 margin을 -4로 두게 되면 기존 groove에서 둔 범위를 초과해서 바가 뜨게 된다.



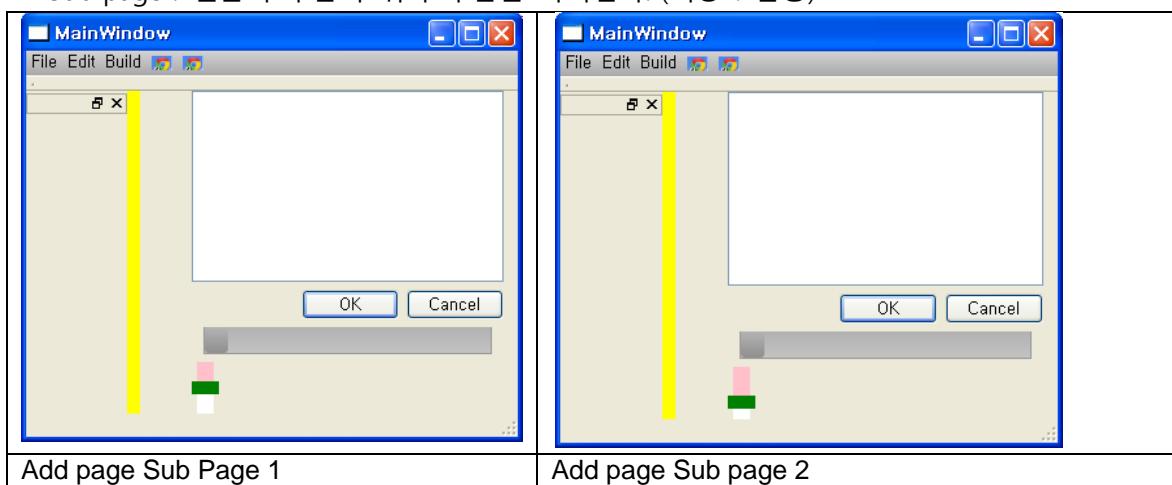
- QSlider::add-page:vertical { background: white;}

- QSlider::sub-page:vertical { background: pink; }

이 부분은 스크롤 바의 상태에 따라서 색상을 변하게 할 수 있는 부분이다.

Add page : 핸들러 부분의 아래 부분을 가리킨다. (색상 : 흰색)

Sub page : 핸들러 부분의 위쪽 부분을 가리킨다. (색상 : 분홍)



19. QSpinBox

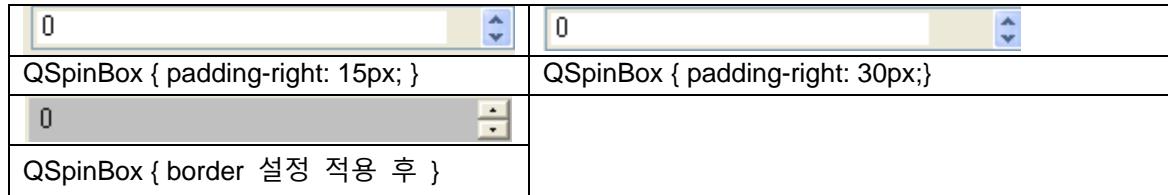
- QSpinBox { padding-right: 15px; border-image: url(:/new/prefix1/frame.png); border-width:3 }

Spinbox는 숫자 카운트 올리고 내리고 하는 박스인 것 같다. 이에 대한 디자인을 할 수 있다.

Padding-right 설정을 하는 이유는 옆에 있는 화살표 공간을 두기 위해서이다. 15px를 준다.

Border-image는 적용을 하게 되면 스피너 테두리 안에 있는 모든 영역이 같은 색으로 덮이게 된다. Frame.png를 사용해서 회색이 되었다.

Border-width는 모서리 부분을 지름 3px 인 원으로 해서 둥글게 표현했다.



- QSpinBox::up-button{ subcontrol-origin:border; subcontrol-position:top right; width:16px;

border-image:url(:/new/prefix1/spinup.png); border-width:1px;}

Spinbox의 버튼 중 up-button 디자인 설정이다. Subcontrol-origin으로 border만큼의 여백을 지니게 한다. 그리고 subcontrol-position을 통해 오른쪽 상단에 위치하게 한다. 너비는 16px로 잡았고 이미지는 인터넷에 구한 이미지를 사용해서 추가했다. Border-width로 모서리 부분을 약간 둥글게 했다.



- QSpinBox::up-button:hover{ border-image:url(:/new/prefix1/.png) 1;}

- QSpinBox::up-button:pressed{ border-image:url(:/new/prefix1/.png) 1;}

위와 같은 up-button에 걸린 디자인 설정이다. Hover는 마우스를 위에 올렸을 때의 모습을 변화 시킬 때 사용하고, pressed는 마우스를 눌렀을 때 변화를 나타낼 때 사용한다.

- QSpinBox::up-arrow {image: url(:/new/prefix1/radiobutton_checked_pressed.png); width:7px; height:7px;}

마지막으로 위쪽 화살표 디자인을 추가할 수 있다. Image를 사용하고 그 너비와 높이는 width와 height로 설정해준다. 편의상 라디오버튼 pressed로 대체했고 아래는 그 예시다.



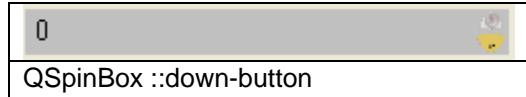
- QSpinBox::up-arrow:disabled, QSpinBox::up-arrow:off { image: url(:/new/prefix1/disabled.png);}

SpinBox의 위쪽 화살표가 비활성화 되어있거나 Max값에 도달했을 경우에 사용되는 아이콘을 설정할 수 있다.

- QSpinBox::down-button{ subcontrol-origin:border; subcontrol-position:bottom right; width:16px; border-image:url(:/new/prefix1/spindown.png); border-width:1px; border-top-width: 0;}

Spinbox의 버튼 중 down-button 디자인 설정이다. Subcontrol-origin으로 border만큼의 여백을 지니게 한다. 그리고 subcontrol-position을 통해 오른쪽 하단에 위치하게 한다. 너비는 16px로 잡

았고 이미지는 인터넷에 구한 이미지를 사용해서 추가했다. Border-width로 모서리 부분을 약간 둑글게 했다. 대신 하단 오른쪽 부분의 테두리 중 상단부분은 upbutton 하단과 중복되기 때문에 border-top-width: 0;으로 하여 중복으로 인해 진하게 표기 될 수도 있는 부분을 억제한다.



- QSpinBox::down-button:hover{ border-image:url(:/new/prefix1/.png) 1;}
- QSpinBox::down-button:pressed{ border-image:url(:/new/prefix1/.png) 1;}

위와 같은 down-button에 걸린 디자인 설정이다. Hover는 마우스를 위에 올렸을 때의 모습을 변화 시킬 때 사용하고, pressed는 마우스를 눌렀을 때 변화를 나타낼 때 사용한다. (결과는 위의 up-button)에서와 같기에 화면은 생략했다.

- QSpinBox::down-arrow {image: url(:/new/prefix1/radiobutton_checked_pressed.png); width:7px; height:7px;}

아래쪽 화살표 디자인을 추가할 수 있다. Image를 사용하고 그 너비와 높이는 width와 height로 설정해준다. 편의상 라디오버튼 pressed로 대체했고 아래는 그 예시다.

위 그림에서 화살표와 겹쳐져있는 RadioButton그림이 있다. 이것이 아래쪽에도 똑같이 생성되게 된다.

- QSpinBox::down-arrow:disabled, QSpinBox::down-arrow:off { image: url(:/new/prefix1/disabled.png);}
- SpinBox의 아래쪽 화살표가 비활성화 되어있거나 Min값에 도달했을 경우에 사용되는 아이콘을 설정할 수 있다.

20. QSplitter

Splitter은 여러 개 QTextEdit과 같은 창의 크기를 조절할 수 있는 장치이다.

```
QTextEdit *editor1 = new QTextEdit;
QTextEdit *editor2 = new QTextEdit;
QTextEdit *editor3 = new QTextEdit;
QSplitter *splitter = new QSplitter(parent);
splitter->addWidget(editor1);
splitter->addWidget(editor2);
splitter->addWidget(editor3);
splitter->setStyleSheet(" QSS 해당 구문을 넣으면 된다.");
splitter->show();
```

여기서 splitter->setStyleSheet("")를 안에 QSS구문을 넣어서 디자인을 할 수 있다.

– QSplitter::handle { image: url(/new/prefix1/stylesheets-vline.png);}

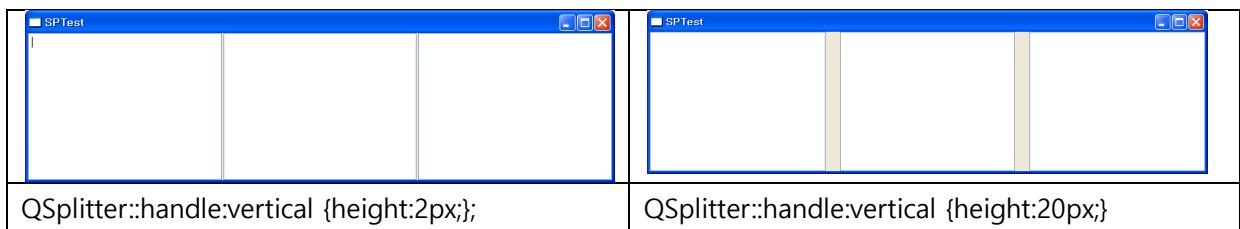
TextEdit 사이의 공간에 image를 넣어서 splitter를 표현할 수 있는 부분이다.



– QSplitter::handle:horizontal { width:2px; }

– QSplitter::handle:vertical { height:2px; }

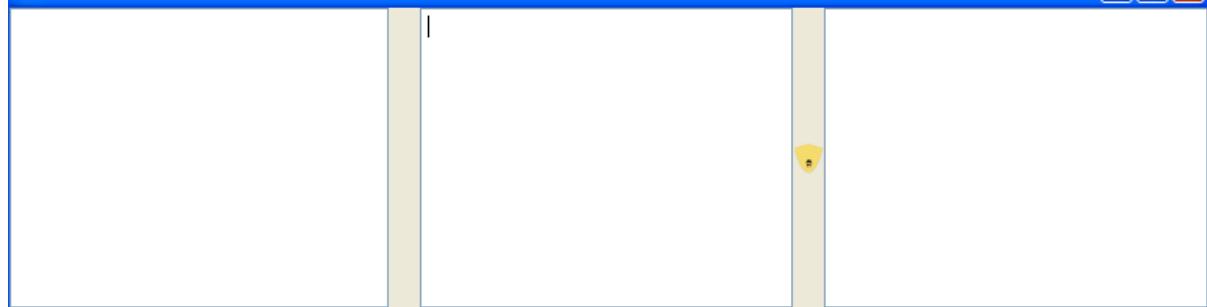
구분선이 세로가 필요하면 vertical, 가로가 필요하면 horizontal을 사용하고, 그 Splitter의 너비를 설정할 수 있다.



QSplitter::handle:vertical {height:2px;}

QSplitter::handle:vertical {height:20px;}

– QSplitter::handle:pressed { image: url(/new/prefix1/spindown.png); }



마우스가 눌러졌을 때 Splitter 핸들러에 나타나는 디자인이다. 여기선 그림을 넣었다.

21. QStatusBar

- StatusBar 는 상태표시줄로써 현재의 작업 상황을 보여주는 메시지 라인이다. 먼저 StatusBar 를 사용하기 위해서는 코딩으로 ui 를 디자인해야 한다. 먼저 dialog.h(헤더파일)에서 QStatusBar 클래스와 그에 대한 's'라는 객체를 생성한다.

```
class QStatusBar;  
  
QStatusBar* s;  
  
#ifndef DIALOG_H  
#define DIALOG_H  
  
#include <QDialog>  
  
namespace Ui {  
    class Dialog;  
}  
class QMenu;  
class QStatusBar;  
  
class Dialog : public QDialog  
{  
    Q_OBJECT  
  
public:  
    explicit Dialog(QWidget *parent = 0);  
    ~Dialog();  
  
private:  
    Ui::Dialog *ui;  
    QMenu* a;  
    QStatusBar* s;  
  
//public Q_SLOTS:  
//    void func();  
};  
  
#endif // DIALOG_H
```

- 's'라는 객체를 이용해서 함수를 사용한다. 먼저 setFixedHeight(30) 상태표시줄의 세로길이를 30 으로 고정시키는 함수이다. 이렇게 고정된 상태에서 setStyleSheet 를 통해 상태표시줄의 디자인을 변경할 수 있다.

```
#include <QStatusBar>  
s = new QStatusBar(this);  
s->setFixedHeight(30);  
s->setStyleSheet("QStatusBar {background: brown; border: 1px solid green;}");  
ui->verticalLayout->addWidget(s);
```

- QStatusBar {background: brown; border: 1px solid green;} : 현재 StatusBar 의 배경색은 갈색이고, border(바깥선)의 굵기는 1px, 선의 종류는 solid, 색상은 초록색으로 설정하고 있다.

```
s->setStyleSheet("QStatusBar {background: brown; border: 1px solid green;}");
```

- QStatusBar::item {border:1px solid red; border-radius: 3px;} : StatusBar(상태표시줄) 안에 있는 item(아이템, 예를 들면 pushbutton)의 디자인을 변경한 것이다. 먼저 border 의 굵기는 1px, 선의 종류는 solid, border-radius(양 쪽 끝의 반지름)은 3px 로 설정하였다.

이전에 앞서 QPushButton(푸쉬버튼)을 추가하기 위해서는 다음과 같은 명령을 해주어야 한다.

```
s->addWidget(new QPushButton("test"));
```

그런 다음에 위와 같은 디자인 변경을 하면 다음과 같이 나타난다.



22. QTabWidget

TabWidget 이란 Tab 들을 Widget 과 결합하여 여러가지 항목들을 모아놓은 것을 말한다.

- **QTabWidget::pane {border-top: 2px solid #C2C7CB;}** : 먼저 TabWidget 의 pane 은 아래 그림과 같은 회색선을 의미하는 것으로 border(바깥선) 굵기를 2px 를 top(위쪽)에 위치시키고, 선의 종류는 solid, 색상은 밝은 회색으로 설정하고 있다.



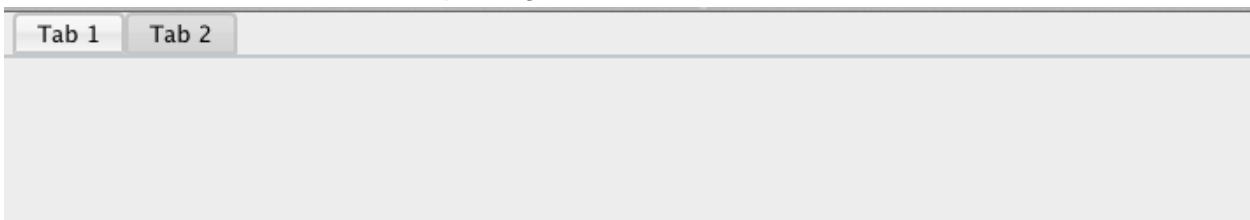
- **QTabWidget::tab-bar {left: 5px;}** : Tab-bar 를 왼쪽으로 5px 만큼 이동시키고 있다.



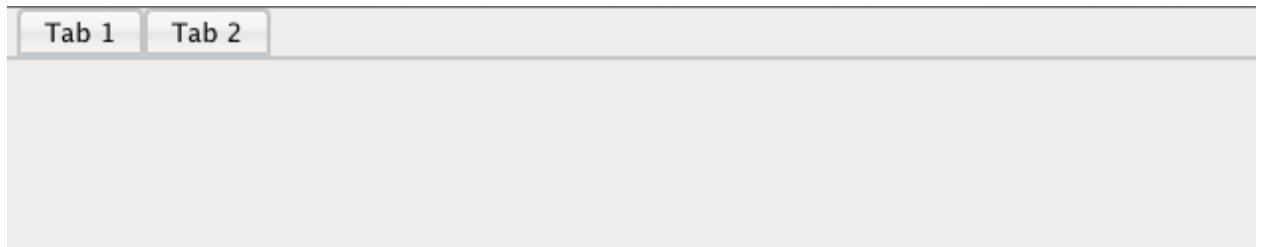
- **QTabBar::tab {background: qlineargradient(x1: 0, y1: 0, x2: 0, y2: 1, stop: 0 #E1E1E1, stop: 0.4 #DDDDDD, stop: 0.5 #D8D8D8, stop: 1.0 #D3D3D3); border: 2px solid #C4C4C3; border-bottom-color: #C2C7CB; border-top-left-radius: 4px; border-top-right-radius: 4px; min-width: 8ex; padding: 2px;}** : 먼저 qlineargradient 를 통해서 그라데이션 효과를 주고 있다. border(바깥선)의 굵기는 2px, 색상은 #C4C4C3, border(바깥선)의 아래쪽 색상은 #C2C7CB, border(바깥선)의 왼쪽 아래 끝의 radius(반지름)은 4px, border(바깥선)의 오른쪽 위 끝의 radius(반지름)은 최소 8ex 가 됨을 보여주고 있다. 마지막으로 이 tab 의 padding(여백공간)은 2px 로 설정되어야 한다.



- **QTabBar::tab:selected, QTabBar::tab:hover{background: qlineargradient(x1: 0, y1: 0, x2: 0, y2: 1, stop: 0 #fafafa, stop: 0.4 #f4f4f4, stop: 0.5 #e7e7e7, stop: 1.0 #fafafa);}** : tab 을 선택했을 때, 그리고 tab hover 효과를 일으키게 하는 설정으로 앞서 나온 hover 는 계속 설명하였듯이 마우스 포인터를 어떤 지점에 갖다 놓았을 때 나타나는 효과이다. 따라서 탭을 마우스로 선택(클릭)하였거나 마우스 포인터를 본인이 원하는 tab 에다 갖다 놓으면 tab 의 배경색상은 다음과 같이 qlineargradient 에 의해 그라데이션 효과가 나타나게 된다.

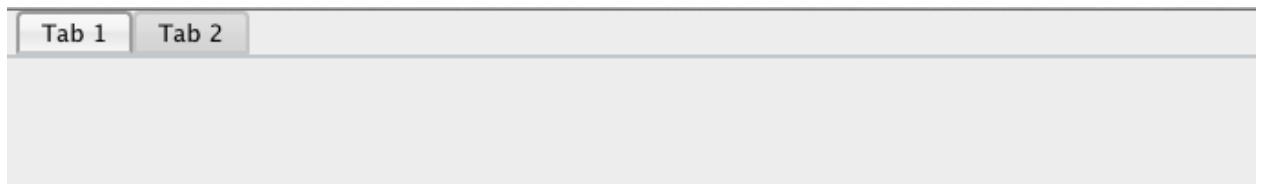


Tab1 을 선택(클릭)하였을 때



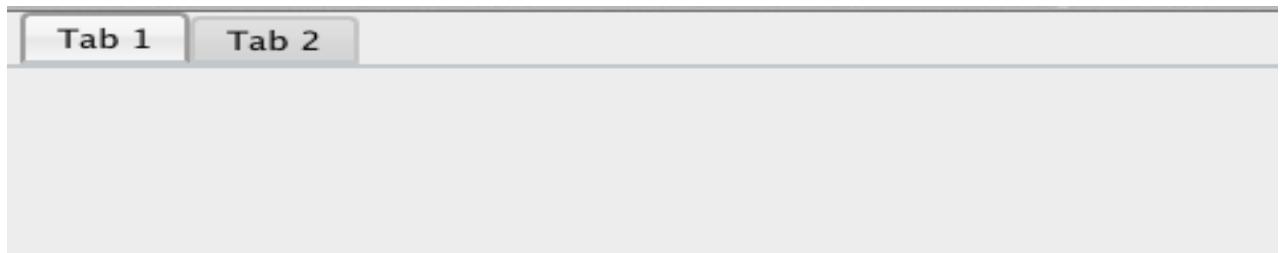
Tab2 에다 마우스 포인터를 갖다 놓았을 때

- `QTabBar::tab:selected {border-color: #9B9B9B; border-bottom-color: #C2C7CB;}` : tab 을 선택(클릭)했을 때의 border(바깥선) 색상을 #9B9B9B, border 의 아래쪽 색상을 #C2C7CB 로 설정하고 있다.



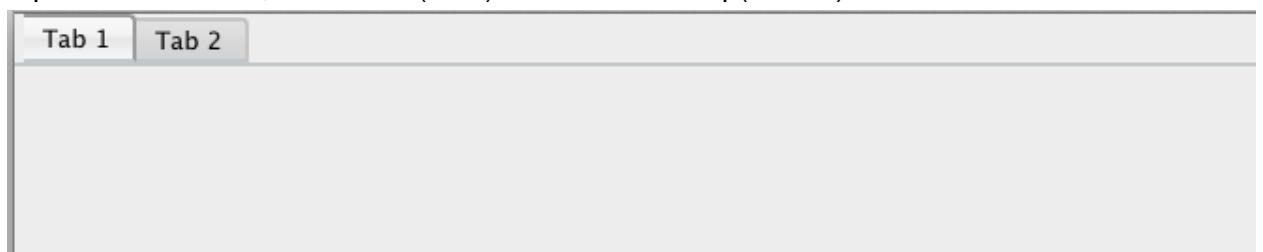
현재 Tab1 을 선택함

- `QTabBar::tab:!selected {margin-top: 2px;}` : tab 을 선택(클릭)하지 않았을 때의 border(바깥선)의 위쪽에 margin(여백공간)을 2px 로 설정함으로써, 선택(클릭)했을 때와 선택(클릭)하지 않았을 때의 차별을 보여주고 있다.



현재 Tab1 을 선택했고, Tab2 는 선택하지 않았음

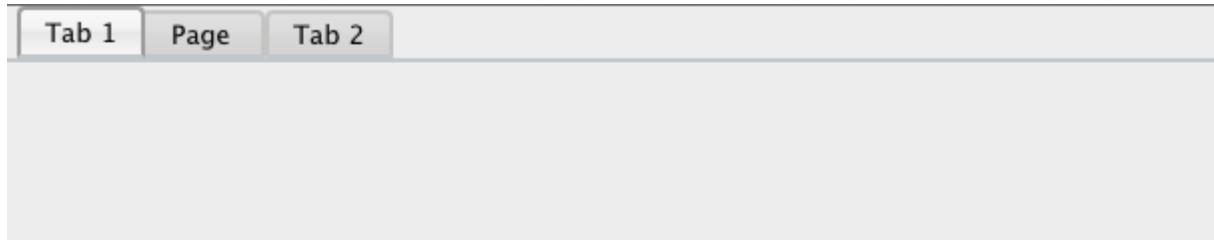
- `QTabBar::tab:selected {margin-left: -4px; margin-right: -4px;}` : 좌우의 margin(여백공간)을 -4px 로 줄임으로써, 탭을 선택(클릭)하게 되면 overlap(오버랩) 효과가 나타난다.



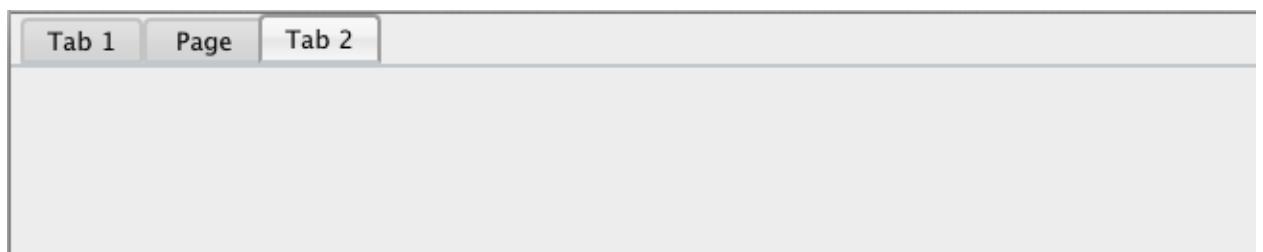
Tab1 을 선택했을 때 overlap(오버랩) 효과가 나타남

- `QTabBar::tab:first:selected {margin-left: 0;}` : 첫 번째 tab 즉, Tab1 의 왼쪽 margin(여백공간)을 없애줌으로써, Tab1 을 선택(클릭)했을 때 아래 그림과 같은 효과를

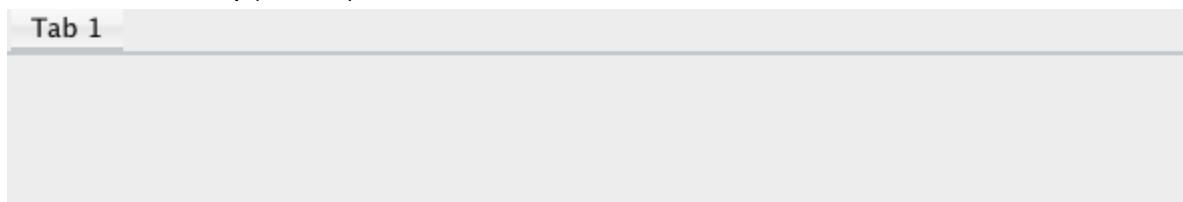
나타내고 있다.



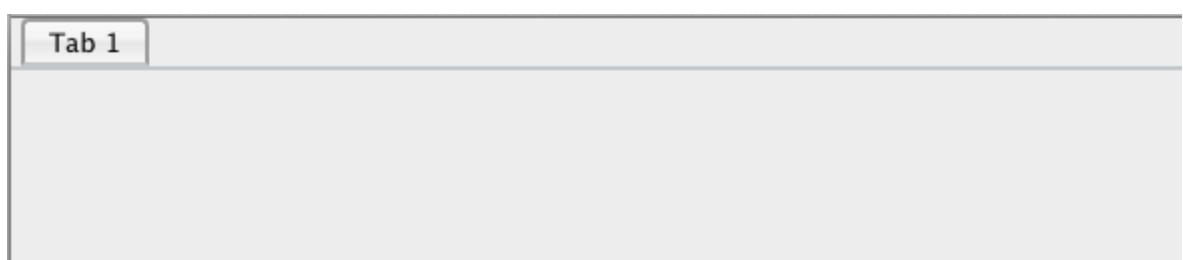
- **QTabBar::tab:last:selected {margin-right: 0;}** : 마지막 세 번째 tab 즉, Tab2 의 오른쪽 margin(여백공간)을 없애줌으로써, Tab2 를 선택(클릭)했을 때 아래 그림과 같은 효과를 나타내고 있다.



- **QTabBar::tab:only-one {margin: 0;}** : 만약 tab 이 하나만 있을 경우에는 margin(여백공간) 을 이용한 overlap(오버랩)효과를 나타내지 않을 것이라는 설정이다.



margin: 0 으로 설정하지 않았을 때

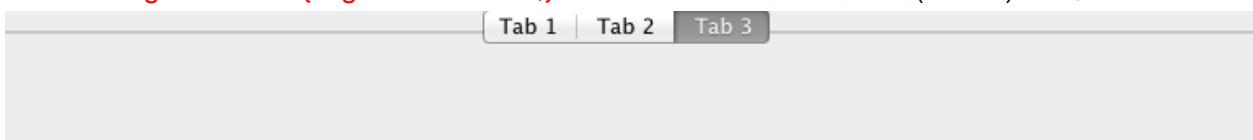


margin: 0 으로 설정하였을 때

- **QTabWidget::pane {border-top: 2px solid #C2C7CB; position: absolute; top: -0.5em;}** : pane 은 아래 그림과 같은 회색 선을 의미하는 것으로 border(바깥선)의 굵기는 2px, 색상은 #C2C7CB, position 은 절대위치, 위로 0.5em 만큼 올라간다.



- QTabWidget::tab-bar {alignment: center;} : tab-bar 의 위치를 가운데(center)로 정렬한다.



- QTabBar::tab {background: qlineargradient(x1: 0, y1: 0, x2: 0, y2: 1, stop: 0 #E1E1E1, stop: 0.4 #DDDDDD, stop: 0.5 #D8D8D8, stop: 1.0 #D3D3D3); border: 2px solid #C4C4C3; border-bottom-color: #C2C7CB; border-top-left-radius: 4px; border-top-right-radius: 4px; min-width: 8ex; padding: 2px;} : 먼저 qlineargradient 를 통해서 그라데이션 효과를 주고 있다. border(바깥선)의 굵기는 2px, 색상은 #C4C4C3, border(바깥선)의 아래쪽 색상은 #C2C7CB, border(바깥선)의 왼쪽 아래 끝의 radius(반지름)은 4px, border(바깥선)의 오른쪽 위 끝의 radius(반지름)은 최소 8ex 가 됨을 보여주고 있다. 마지막으로 이 tab 의 padding(여백공간)은 2px 로 설정되어야 한다.



- QTabBar::tab:selected, QTabBar::tab:hover{background: qlineargradient(x1: 0, y1: 0, x2: 0, y2: 1, stop: 0 #fafafa, stop: 0.4 #f4f4f4, stop: 0.5 #e7e7e7, stop: 1.0 #fafafa);} : tab 을 선택했을 때, 그리고 tab hover 효과를 일으키게 하는 설정으로 앞서 나온 hover 는 계속 설명하였듯이 마우스 포인터를 어떤 지점에 갖다 놓았을 때 나타나는 효과이다. 따라서 템을 마우스로 선택(클릭)하였거나 마우스 포인터를 본인이 원하는 tab 에다 갖다 놓으면 tab 의 배경색상은 다음과 같이 qlineargradient 에 의해 그라데이션 효과가 나타나게 된다.

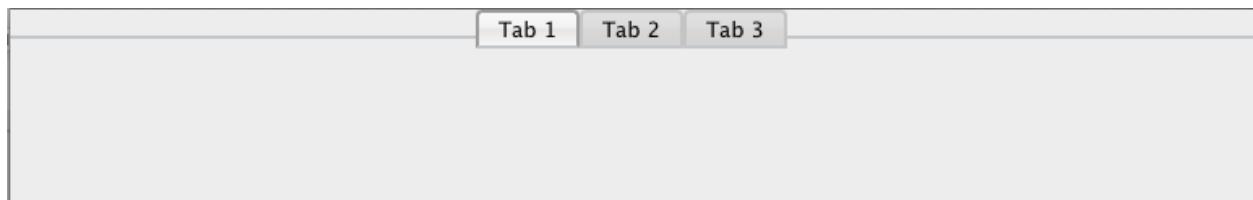


Tab 1 을 선택(클릭)했을 때



Tab 2 를 선택(클릭) 했을 때

- QTabBar::tab:selected {border-color: #9B9B9B; border-bottom-color: #C2C7CB;} : tab 을 선택(클릭)했을 때의 border(바깥선) 색상을 #9B9B9B, border 의 아래쪽 색상을 #C2C7CB 로 설정하고 있다.



Tab 1 을 선택(클릭) 하였을 때

- QTabBar::tear {image:url(:/new/prefix1/checked_hover.png);} : [미해결] 화면에 이미지 파일이 표시되지 않음

- QTabBar::scroller{width: 20px;} : [미해결] 화면에 표시되지 않음

- QTabBar QToolButton {border-image: url(:/new/prefix1/scroll_button.png);} : [미해결] 화면에 표시되지 않음

- QTabBar QToolButton::right-arrow {image: url(:/new/prefix1/right-arrow.jpeg);} : [미해결] 화면에 표시되지 않음

- QTabBar QToolButton::left-arrow {image: url(:/new/prefix1/left-arrow.jpg);} : [미해결] 화면에 표시되지 않음

- QTabBar::close-button {image: url(:/new/prefix1/close_button.svg.hi.png);} : [미해결] 화면에 표시되지 않음

- QTabBar::close-button:hover {image: url(:/new/prefix1/checked_press.jpeg);} : [미해결] 화면에 표시되지 않음

23. QTableView

ui에서 TableView를 만든 다음 mainwindow.cpp에서 다음과 같은 선언을 하면서 Table을 만들 어준다.

```
QStandardItemModel *model = new QStandardItemModel(2,3,this); // 2rows 3 columns
```

model이라는 ItemModel을 만들고 2행 3열의 Table을 선언해준다.

```
model->setHorizontalHeaderItem(0, new QStandardItem(QString("Day")));
model->setHorizontalHeaderItem(1, new QStandardItem(QString("Date")));
model->setHorizontalHeaderItem(2, new QStandardItem(QString("Event")));
```

HorizontalHeaderItem을 이용하면 제일 위의 항목들을 문자열로 설정할 수 있다.

```
QStandardItem *firstRow0 = new QStandardItem(QString("Monday"));
QStandardItem *firstRow1 = new QStandardItem(QString("25 June"));
QStandardItem *firstRow2 = new QStandardItem(QString("Meeting"));
QStandardItem *secondRow0 = new QStandardItem(QString("Tuesday"));
QStandardItem *secondRow1 = new QStandardItem(QString("26 June"));
QStandardItem *secondRow2 = new QStandardItem(QString("Appointment"));
```

내용부분은 StandardItem을 이용해서 문자열들을 만들어주고 model에 삽입한다.

```
model->setItem(0,0,firstRow0);
model->setItem(0,1,firstRow1);
model->setItem(0,2,firstRow2);
model->setItem(1,0,secondRow0);
model->setItem(1,1,secondRow1);
model->setItem(1,2,secondRow2);
```

삽입 코드

```
ui->tableView->setModel(model);
```

ui에서 만든 tableView에 만들어준 모델을 적용해준다.

```
ui->tableView->setStyleSheet("selection-background-color: qlineargradient(x1: 0, y1: 0, x2: 0.5, y2: 0.5, stop: 0 #FF92BB, stop: 1 white);");
```

선택한 Table 부분의 색상을 #FF92BB -> white로 사각형의 절반만큼 그라데이션 해준다.



```
ui->tableView->setStyleSheet("QTableView {selection-background-color: qlineargradient(x1: 0, y1: 0, x2: 0.5, y2: 0.5, stop: 0 #FF92BB, stop: 1 white);} QTableCornerButton::section {background: red; border: 2px outset red;}");
```

위에서 바꾼 색상 적용과 더불어 제일 코너 쪽의 디자인을 할 수 있는 부분이다.

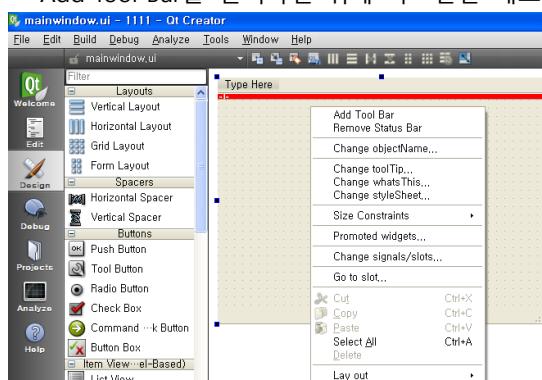
QTableCornerButton::section은 코너 버튼 영역의 스타일을 정한다는 의미. 빨간색으로 배경을 정한다. 테두리는 2px이며 outset 설정으로 돌출선이라는 의미를 가지고 있다.



24. QToolBar

- MainWindow에서 우클릭을 하게되면 툴바를 만들 것인지에 대한 선택창이 나온다.

Add Tool Bar를 선택하면 위에 자그만한 네모의 툴바가 만들어지게 된다.



- QToolBar { background: red; spacing: 3px; }

툴바 영역에 대한 설정을 할 수 있는 곳이다. 여기서 background는 툴바 배경색을 의미하게되고 적용된 것은 위에서도 확인할 수 있다.

Spacing은 툴바 간격을 의미하는 것 같은데 수치를 변경해도 큰 변화는 없었다.. 안에 컨텐츠를 추가하면 변화가 있을지도 모르겠다.

- QToolBar::handle { image: url(:/new/prefix1/icon1.png); }



툴바 안의 각각의 내용에 대한 변경인 것 같은데 여러 개의 툴바를 만들면 일괄 적용되는게 QSS에서의 한계인 것 같은 느낌이 든다. 해결방법을 찾아봤으나 찾기가 어려웠다. 각 툴바마다 삽입한 이미지로 변한다.

25. QToolBox

- 툴박스는 살펴본 결과로는 글을 접었다 펼치는 것처럼 페이지를 나눠서 보고싶은 페이지를 열면 나머지는 접히는 것 같다.

- `QToolBox::tab { background: qlineargradient(x1:0, y1:0, x2:0, y2:1, stop: 0 #E1E1E1, stop: 0.4 #DDDDDD, stop: 0.5 #D8D8D8, stop: 1.0 #D3D3D3); border-radius: 5px; color: darkgray; }`

툴박스의 탭 영역에 대한 디자인을 할 수 있다. 예시 그림에선 Page1, Page2 영역이 이에 해당된다. 배경은 `qlineargradient` 함수를 통해 그라데이션을 써서 처리했고, `border-radius:5px`로 모서리를 둥글게 처리했다. 그리고 색상을 어두운 회색으로 디자인을 마무리했다.

- `QToolBox::tab:selected { font:italic; color:white;}`

탭 영역 중에서 선택한 영역에 대한 디자인을 나타낼 때 쓰는 방식이다. 여기서는 폰트를 이탤릭으로 기울였고, 색상을 흰색으로 해서 기존의 색과 구별할 수 있게 했다.



26. QToolButton

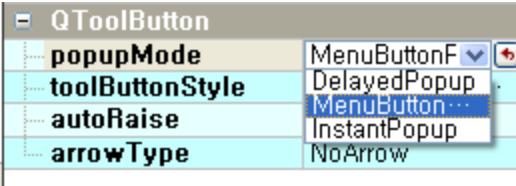
- `QToolButton { border: 2px solid #8f8f91; border-radius: 6px; background-color: qlineargradient(x1:0, y1:0, x2: 0, y2:1, stop 0 #f6f7fa, stop: 1 #dadbd);}`

모든 타입의 툴 버튼에 대한 디자인이다. 여기서 테두리 선은 2px에 `#8f8f91` 색상이고 테두리 선은 6px 지름을 가진 둥근 모양으로 모서리 부분이 처리되었다. 그리고 배경색은 `qlineaergradient` 함수를 통해 그라데이션 되었다.



```
- QToolButton[popupMode="1"] { padding-right: 20px; }
```

QToolButton 옆에 설정에서 toolButtonStyle에서 MenuButtonPopup을 선택해준다.



popupMode=1인 경우 글씨에 오른쪽부분의 여백을 20px 둔다는 의미이다.



```
- QToolButton::pressed { background-color:qlineargradient(x1:0, y1:0, x2: 0, y2:1, stop 0 #dadbdbe, stop: 1 #f6f7fa);}
```

버튼이 눌러졌을 때 디자인이 바뀌는 것이다. 배경 색상을 바꾸면서 우리가 평소에 버튼을 누를 때 보던 디자인을 볼 수 있다.



```
- QToolButton::menu-button { border: 2px solid gray; border-top-right-radius: 6px; border-bottom-right-radius: 6px; width: 16px;}
```

위의 메뉴버튼 팝업모드일 때 옆에 화살표 버튼을 디자인 할 수 있는 QSS 구문이다.

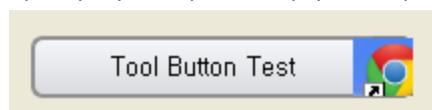
테두리 선은 2px 실선 회색이고, 오른쪽 상단, 오른쪽 하단의 모서리를 둥글게 처리하면서 너비를 16px로 잡았다.



- QToolButton::menu-arrow { image: url(/new/prefix1/icon1.png);}

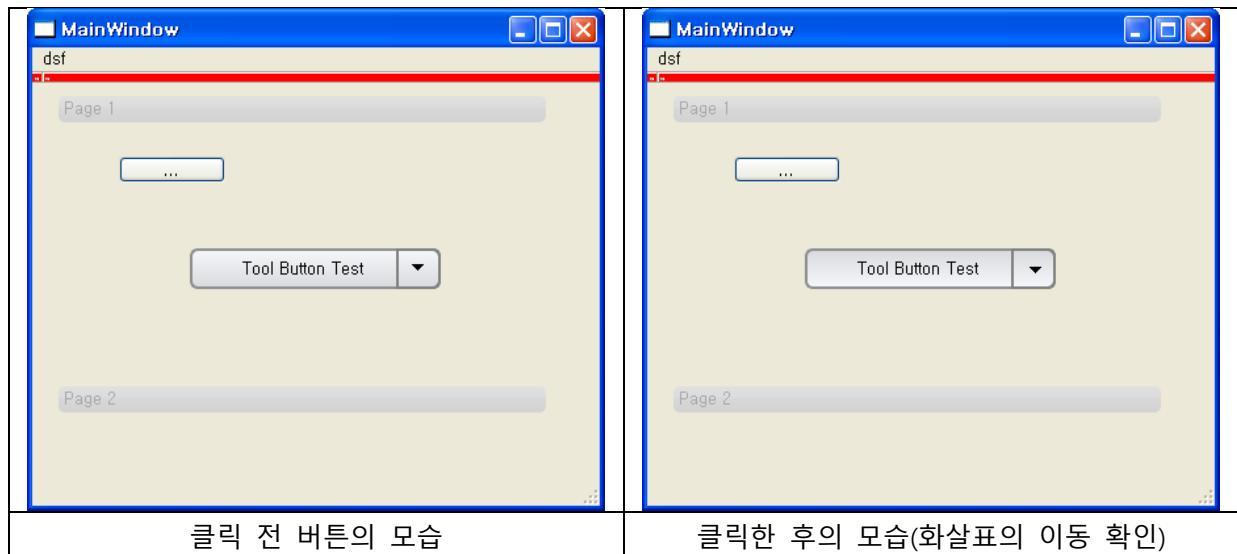
QToolButton	
popupMode	MenuButtonPopup
toolButtonStyle	ToolButtonIcon...
autoRaise	<input type="checkbox"/>
arrowType	NoArrow

사용하려면 ToolButton의 모드를 MenuButtonPopUp으로 바꾼 다음 이용하면 된다. 그러면 메뉴 버튼의 화살표부분을 디자인할 수 있다. 여기서는 그림파일로 대체했다.



. - QToolButton::menu-arrow:open { top: 1px; left: 1px;}

MenuButtonPopUp모드인 ToolButton을 클릭했을 때 오른쪽에 있는 화살표를 누른 것과 같은 효과를 주기 위해 위에 여백 1px 왼쪽 여백 1px를 줘서 약간 이동 시키는 방법이다.



27. QToolTip

마우스를 위에 올렸을 때 뜨는 설명문과 같은 기능이다. 이를 QSS에서 디자인할 수 있다.

. - QToolTip { border:2px solid darkkhaki; padding: 5px; border-radius: 3px; opacity: 100; }

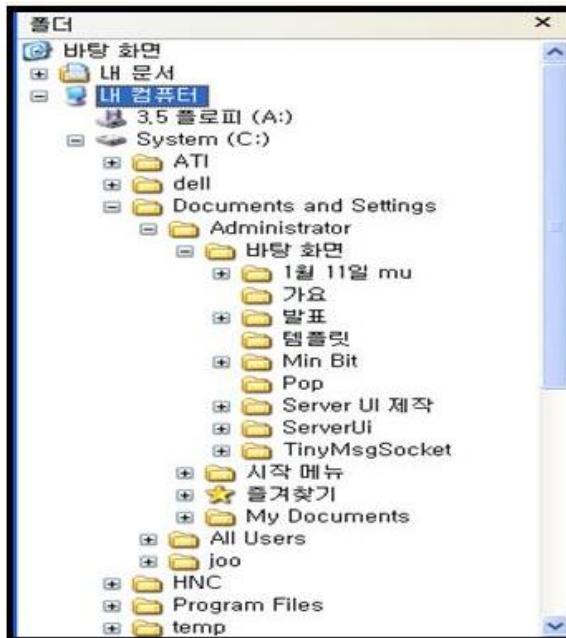
테두리는 2px 실선 darkkhaki 색상, padding(내부 간격) 5px 모서리 둥글게 처리 3px

Opacity(불투명도) : 100 (숫자가 높을수록 불투명해짐)



28. TreeView

TreeView 란 계층적인 구조를 가지는 자료를 표시하는 컨트롤이며 탐색기의 왼쪽에 있는 디렉터리 트리가 대표적이라고 할 수 있다. 사용자 하드 디스크의 모든 서브 디렉터리는 물론 네트워크 자원까지 한눈에 체계적으로 보여준다.



TreeView 의 구조

먼저 디자인을 하기에 앞서 `treeView` 를 사용하기 위해서는 다음과 같은 설정을 해주어야 한다.

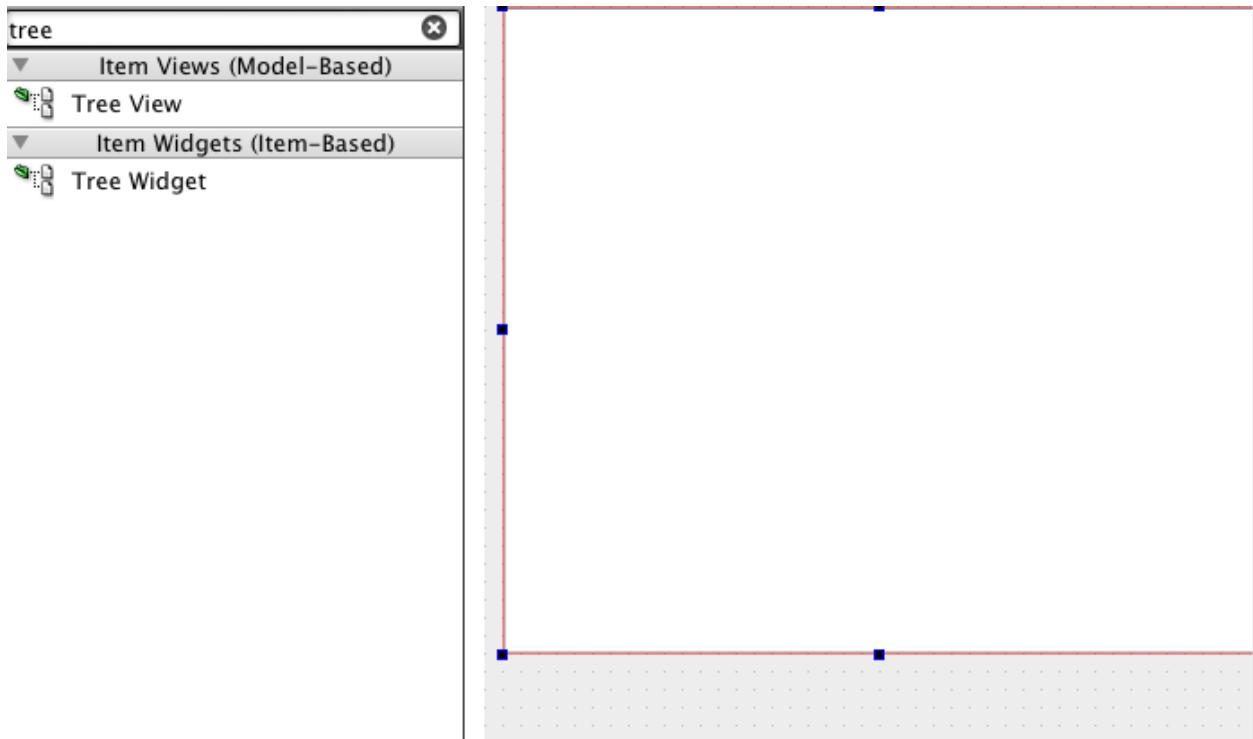
`dialog.h` 파일에서 아래 그림과 같이 `#include` 를 통해 `<QtCore>`, `<QtGui>`, `<QDirModel>`을 입력한다.

```
#include <QtCore>
#include <QtGui>
#include <QDirModel>
```

위와 같이 입력을 한 뒤에 `Dialog` 클래스 안에 있는 `private` 영역 안에서 `QDirModel`이라는 타입을 가진 `model` 객체를 생성해준다.

```
QDirModel *model;
```

`model`이라는 객체를 생성해 준 뒤에 `dialog.ui`에서 `treeView` 를 직접 넣어준다.



이렇게 넣어준 `treeView` 를 큰 틀이라고 본다면 이제 이 안에 다음과 같이 `model` 을 넣어주면 된다. 먼저 `dialog.cpp` 파일로 넘어가서 생성한 `model` 객체를 가지고 Dialog 클래스 안에 있는 생성자 함수(`Dialog::Dialog(QWidget *parent)` : `QDialog(parent), ui(new Ui::Dialog){~~~ }`)로 들어가서 `model = new QDirModel(this);`라는 선언을 해주고 `model` 이 읽는 것만 가능하게 하는 것을 방지하기 위해 `setReadOnly(false);`를 선언해 준다. 그리고 마지막으로 아래 그림과 같이 `treeView` 모델을 장착해 준다.

```
model = new QDirModel(this);
model->setReadOnly(false);

ui->treeView->setModel(model);
```

이 모든 세팅이 끝나면 다음 예제를 가지고 디자인 작업을 시작한다.

Name	Size	Kind	Date Modified
▼ /	--	Drive	13. 3. 30. 오전 10:15
► A...	--	Folder	14. 1. 10. 오후 4:45
► ...	--	Folder	14. 1. 9. 오전 9:44
► L...	--	Folder	14. 1. 9. 오전 9:45
► S...	--	Folder	13. 3. 30. 오후 3:02
► ...	--	Folder	12. 3. 9. 오후 6:04
► ...	--	Folder	12. 3. 9. 오후 6:04

초기 디자인 화면

- QTreeView {alternate-background-color: yellow;} : treeView 의 배경색상을 노란색으로 대체시킨다.

- QTreeView {show-decoration-selected: 1;} : 선택 항목 장식을 하나만 하겠다는 의미

- QTreeView::item {border: 1px solid #d9d9d9; border-top-color: transparent; border-bottom-color: transparent;} : item(항목)을 하나 선택하면 border(바깥선)의 굵기는 1px, 선의 종류는 solid, 색상은 #d9d9d9, border(바깥선)의 위와 아래는 아래 화면과 같이 투명해진다.

Name	Size	Kind	Date Modified
▼ /	--	Drive	13. 3. 30. 오전 10:15
▶ Developer	--	Folder	14. 1. 9. 오전 9:44
▶ Library	--	Folder	14. 1. 9. 오전 9:45
▶ System	--	Folder	13. 3. 30. 오후 3:02
▶ Users	--	Folder	12. 3. 9. 오후 6:04
▶ 사용 설명서와 정보	--	Folder	12. 3. 9. 오후 6:04

‘Applications’ 폴더를 클릭했을 때

- QTreeView::item:hover {background: qlineargradient(x1: 0, y1: 0, x2: 0, y2: 1, stop: 0 #e7effd, stop: 1 #cbdaf1); border: 1px solid #bfcde4;} : item 을 클릭하기 전에 마우스 포인터를 해당 item 에 갖다 놓으면 위와 같이 그라데이션 효과를 입힌 배경색과 1px 굵기, solid, #bfcde4 의 색상을 가진 border(바깥선)의 디자인이 나온다. 실행 화면 결과는 아래와 같다.

Name	Size	Kind	Date Modified
▼ /	--	Drive	13. 3. 30. 오전 10:15
▼ Applications	--	Folder	14. 1. 10. 오후 4:45
▶ ALZip.app	--	Folder	12. 3. 30. 오전 10:31
▶ Address Book.app	--	Folder	12. 10. 29. 오전 9:24
▶ App Store.app	--	Folder	13. 7. 3. 오전 9:30
▶ Automator.app	--	Folder	12. 10. 29. 오전 9:24
▶ Calculator.app	--	Folder	12. 10. 29. 오전 9:24
▶ Chess.app	--	Folder	12. 10. 29. 오전 9:24
▶ Cornerstone.app	--	Folder	13. 3. 30. 오전 10:52
▶ DVD Player.app	--	Folder	12. 10. 29. 오전 9:24
▶ Dashboard.app	--	Folder	12. 10. 29. 오전 9:24
▶ Dictionary.app	--	Folder	12. 10. 29. 오전 9:24
▶ Evernote.app	--	Folder	13. 5. 24. 오전 10:44
▶ FaceTime.app	--	Folder	12. 10. 29. 오전 9:24
▶ Firefox.app	--	Folder	13. 12. 6. 오전 6:51
▶ Font Book.app	--	Folder	12. 10. 29. 오전 9:24
▶ GarageBand.app	--	Folder	12. 3. 20. 오후 2:30
▶ Google Chrome.app	--	Folder	14. 1. 23. 오전 11:48
▶ HanwordViewer.app	--	Folder	13. 5. 20. 오후 1:39
▶ Image Capture.app	--	Folder	12. 10. 29. 오전 9:24

현재 'Address Book.app'에다 마우스 포인터를 갖다 두었더니 위와 같은 효과가 나타났다.

- `QTreeView::item:selected {border: 1px solid #567dbc;}` : item 을 선택했을 때 border(바깥선)의 굵기는 1px 이고, 선의 종류는 solid, 색상은 #567dbc 이다. 실행 화면은 아래와 같다.

Name	Size	Kind	Date Modified
▼ /		-- Drive	13. 3. 30. 오전 10:15
▼ Applications		-- Folder	14. 1. 10. 오후 4:45
▶ ALZip.app		-- Folder	12. 3. 30. 오전 10:31
▶ App Store.app		-- Folder	13. 7. 3. 오전 9:30
▶ Automator.app		-- Folder	12. 10. 29. 오전 9:24
▶ Calculator.app		-- Folder	12. 10. 29. 오전 9:24
▶ Chess.app		-- Folder	12. 10. 29. 오전 9:24
▶ Cornerstone.app		-- Folder	13. 3. 30. 오전 10:52
▶ DVD Player.app		-- Folder	12. 10. 29. 오전 9:24
▶ Dashboard.app		-- Folder	12. 10. 29. 오전 9:24
▶ Dictionary.app		-- Folder	12. 10. 29. 오전 9:24
▶ Evernote.app		-- Folder	13. 5. 24. 오전 10:44
▶ FaceTime.app		-- Folder	12. 10. 29. 오전 9:24
▶ Firefox.app		-- Folder	13. 12. 6. 오전 6:51
▶ Font Book.app		-- Folder	12. 10. 29. 오전 9:24

'Address Book.app'을 클릭하였더니(선택하였더니) 위와 같은 디자인을 볼 수 있다.

마지막으로 위에서 했던 item 항목을 모두 합쳐보겠다. 실행화면은 아래와 같다.

Name	Size	Kind	Date Modified
▼ /		-- Drive	13. 3. 30. 오전 10:15
▼ Applications		-- Folder	14. 1. 10. 오후 4:45
▶ ALZip.app		-- Folder	12. 3. 30. 오전 10:31
▶ App Store.app		-- Folder	13. 7. 3. 오전 9:30
▶ Automator.app		-- Folder	12. 10. 29. 오전 9:24
▶ Calculator.app		-- Folder	12. 10. 29. 오전 9:24
▶ Chess.app		-- Folder	12. 10. 29. 오전 9:24
▶ Cornerstone.app		-- Folder	13. 3. 30. 오전 10:52
▶ DVD Player.app		-- Folder	12. 10. 29. 오전 9:24
▶ Dashboard.app		-- Folder	12. 10. 29. 오전 9:24
▶ Dictionary.app		-- Folder	12. 10. 29. 오전 9:24
▶ Evernote.app		-- Folder	13. 5. 24. 오전 10:44
▶ FaceTime.app		-- Folder	12. 10. 29. 오전 9:24
▶ Firefox.app		-- Folder	13. 12. 6. 오전 6:51

- QTreeView::branch {background: palette(base);} : 하위 항목을 가리키는 가지를 초기화 시키는 것으로 아래 화면의 모습과 같아진다.

Name	Size	Kind	Date Modified
└ A...	--	Folder	14. 1. 10. 오후 4:45
└ ...	--	Folder	14. 1. 9. 오전 9:44
└ L...	--	Folder	14. 1. 9. 오전 9:45
└ S...	--	Folder	13. 3. 30. 오후 3:02
└ ...	--	Folder	12. 3. 9. 오후 6:04
└ ...	--	Folder	12. 3. 9. 오후 6:04

- QTreeView::branch:has-siblings:!adjoins-item {background: cyan;} : 하위 항목을 가리키는 가지의 디자인 모양을 보여주고 있다. 먼저 has-siblings 이란 한 트리(폴더) 안에 동등한 노드(동등한 위치)에 있는 폴더를 의미하고 !adjoins-item 은 인접하지 않은 아이템들을 뜻하는 것으로써, 동등한 위치에 있는 폴더이면서 서로 인접하고 있지 않는 아이템들을 묶고 있다. 여기서 '!'은 not(부정)을 의미한다. 이 가지의 배경색은 청록색이므로 명령을 실행하면 아래의 화면과 같다.

Name	Size	Kind	Date Modified
└ /	--	Drive	13. 3. 30. 오전 10:15
└ Applications	--	Folder	14. 1. 10. 오후 4:45
└ ALZip.app	--	Folder	12. 3. 30. 오전 10:31
└ CodeResources	7 KB	File	12. 3. 30. 오전 10:30
└ Info.plist	6 KB	plist File	12. 3. 30. 오전 10:30
└ MacOS	--	Folder	12. 1. 26. 오후 7:58
└ PkgInfo	8 byte(s)	File	12. 3. 30. 오전 10:30
└ Resources	--	Folder	11. 8. 16. 오후 7:07
└ _CodeSignature	--	Folder	11. 8. 16. 오후 7:07
└ _MASReceipt	--	Folder	12. 3. 30. 오전 10:31
└ Address Book.app	--	Folder	12. 10. 29. 오전 9:24
└ App Store.app	--	Folder	13. 7. 3. 오전 9:30
└ Automator.app	--	Folder	12. 10. 29. 오전 9:24
└ Calculator.app	--	Folder	12. 10. 29. 오전 9:24
└ Chess.app	--	Folder	12. 10. 29. 오전 9:24
└ Cornerstone.app	--	Folder	13. 3. 30. 오전 10:52
└ DVD Player.app	--	Folder	12. 10. 29. 오전 9:24
└ Dashboard.app	--	Folder	12. 10. 29. 오전 9:24
└ Dictionary.app	--	Folder	12. 10. 29. 오전 9:24
└ Evernote.app	--	Folder	13. 5. 24. 오전 10:44
└ FaceTime.app	--	Folder	12. 10. 29. 오전 9:24
└ Firefox.app	--	Folder	13. 12. 6. 오전 6:51
└ Font Book.app	--	Folder	12. 10. 29. 오전 9:24
└ GarageBand.app	--	Folder	12. 3. 20. 오후 2:30
└ Google Chrome.app	--	Folder	14. 1. 14. 오전 12:38
└ HanwordViewer.app	--	Folder	13. 5. 20. 오후 1:39
└ Image Capture.app	--	Folder	12. 10. 29. 오전 9:24
└ Keynote.app	--	Folder	13. 3. 4. 오후 4:45

- QTreeView::branch:has-siblings:adjoins-item {background: red;} : has-siblings 이므로 한 트리(폴더) 안에 동등한 노드(동등한 위치)이면서 adjoins-item 이므로 인접한 폴더만 가지 표시를 해주게 된다. 배경색은 빨간색으로 명령을 실행하면 아래의 화면과 같다.

Name	Size	Kind	Date Modified
/	--	Drive	13. 3. 30. 오전 10:15
Applications	--	Folder	14. 1. 10. 오후 4:45
ALZip.app	--	Folder	12. 3. 30. 오전 10:31
CodeResources	7 KB	File	12. 3. 30. 오전 10:30
Info.plist	6 KB	plist File	12. 3. 30. 오전 10:30
MacOS	--	Folder	12. 1. 26. 오후 7:58
PkgInfo	8 byte(s)	File	12. 3. 30. 오전 10:30
Resources	--	Folder	11. 8. 16. 오후 7:07
_CodeSignature	--	Folder	11. 8. 16. 오후 7:07
_MASReceipt	--	Folder	12. 3. 30. 오전 10:31
Address Book.app	--	Folder	12. 10. 29. 오전 9:24
App Store.app	--	Folder	13. 7. 3. 오전 9:30
Automator.app	--	Folder	12. 10. 29. 오전 9:24
Calculator.app	--	Folder	12. 10. 29. 오전 9:24
Chess.app	--	Folder	12. 10. 29. 오전 9:24
Cornerstone.app	--	Folder	13. 3. 30. 오전 10:52
DVD Player.app	--	Folder	12. 10. 29. 오전 9:24
Dashboard.app	--	Folder	12. 10. 29. 오전 9:24
Dictionary.app	--	Folder	12. 10. 29. 오전 9:24
Evernote.app	--	Folder	13. 5. 24. 오전 10:44
FaceTime.app	--	Folder	12. 10. 29. 오전 9:24
Firefox.app	--	Folder	13. 12. 6. 오전 6:51
Font Book.app	--	Folder	12. 10. 29. 오전 9:24
GarageBand.app	--	Folder	12. 3. 20. 오후 2:30
Google Chrome.app	--	Folder	14. 1. 14. 오전 12:38
HanwordViewer.app	--	Folder	13. 5. 20. 오후 1:39
Image Capture.app	--	Folder	12. 10. 29. 오전 9:24
Keynote.app	--	Folder	13. 3. 4. 오후 4:45

다음의 아래 화면은 이 두 가지의 명령을 합친 것과 같다.

Name	Size	Kind	Date Modified
/	--	Drive	13. 3. 30. 오전 10:15
Applications	--	Folder	14. 1. 10. 오후 4:45
ALZip.app	--	Folder	12. 3. 30. 오전 10:31
CodeResources	7 KB	File	12. 3. 30. 오전 10:30
Info.plist	6 KB	plist File	12. 3. 30. 오전 10:30
MacOS	--	Folder	12. 1. 26. 오후 7:58
PkgInfo	8 byte(s)	File	12. 3. 30. 오전 10:30
Resources	--	Folder	11. 8. 16. 오후 7:07
_CodeSignature	--	Folder	11. 8. 16. 오후 7:07
_MASReceipt	--	Folder	12. 3. 30. 오전 10:31
Address Book.app	--	Folder	12. 10. 29. 오전 9:24
App Store.app	--	Folder	13. 7. 3. 오전 9:30
Automator.app	--	Folder	12. 10. 29. 오전 9:24
Calculator.app	--	Folder	12. 10. 29. 오전 9:24
Chess.app	--	Folder	13. 3. 30. 오전 10:52
Cornerstone.app	--	Folder	12. 10. 29. 오전 9:24
DVD Player.app	--	Folder	12. 10. 29. 오전 9:24
Dashboard.app	--	Folder	12. 10. 29. 오전 9:24
Dictionary.app	--	Folder	12. 10. 29. 오전 9:24
Evernote.app	--	Folder	13. 5. 24. 오전 10:44
FaceTime.app	--	Folder	12. 10. 29. 오전 9:24
Firefox.app	--	Folder	13. 12. 6. 오전 6:51
Font Book.app	--	Folder	12. 10. 29. 오전 9:24
GarageBand.app	--	Folder	12. 3. 20. 오후 2:30
Google Chrome.app	--	Folder	14. 1. 14. 오전 12:38
HanwordViewer.app	--	Folder	13. 5. 20. 오후 1:39
Image Capture.app	--	Folder	12. 10. 29. 오전 9:24
Keynote.app	--	Folder	13. 3. 4. 오후 4:45

- QTreeView::branch:!has-children:!has-siblings:adjoins-item {background: blue;} : !has-

`children` 이므로 자식을 가지지 않는(하위항목을 가지지 않는) 폴더이고 동시에 자신과 동등한 노드(동등한 위치)를 갖지도 않아야 하면서 자신과 인접한 아이템들을 표시해야 한다. 배경색은 파란색이므로 실행 화면은 아래 다음과 같다.

Name	Size	Kind	Date Modified
/	--	Drive	13. 3. 30. 오전 10:15
Applications	--	Folder	14. 1. 10. 오후 4:45
ALZip.app	--	Folder	12. 3. 30. 오전 10:31
Contents	--	Folder	12. 3. 30. 오전 10:31
CodeResources	7 KB	File	12. 3. 30. 오전 10:30
Info.plist	6 KB	plist File	12. 3. 30. 오전 10:30
MacOS	--	Folder	12. 1. 26. 오후 7:58
ALZip	1.2 MB	File	12. 3. 30. 오전 10:30
bin	--	Folder	11. 8. 16. 오후 7:07
7z.so	2.2 MB	so File	12. 3. 30. 오전 10:30
PkgInfo	8 byte(s)	File	12. 3. 30. 오전 10:30
Resources	--	Folder	11. 8. 16. 오후 7:07
Resources			
02.png	13 KB	png File	11. 8. 16. 오후 7:07
03.png	13 KB	png File	11. 8. 16. 오후 7:07
04.png	13 KB	png File	11. 8. 16. 오후 7:07
05.png	13 KB	png File	11. 8. 16. 오후 7:07
06.png	13 KB	png File	11. 8. 16. 오후 7:07
07.png	13 KB	png File	11. 8. 16. 오후 7:07
08.png	14 KB	png File	11. 8. 16. 오후 7:07
09.png	14 KB	png File	11. 8. 16. 오후 7:07
10.png	13 KB	png File	11. 8. 16. 오후 7:07
11.png	14 KB	png File	11. 8. 16. 오후 7:07
12.png	13 KB	png File	11. 8. 16. 오후 7:07

현재 위의 화면을 보면 '7z.so'파일은 하위항목을 가지지 않고 동시에 자신과 동등한 노드에 아무런 파일이 없으며 자신과 인접한 아이템을 표시하고 있다.

- `QTreeView::branch:closed:has-children:has-siblings {background: pink;}` : closed 는 '닫혀 있다.'라는 뜻으로 현재 열지 않은 폴더를 의미한다. 따라서 현재 열지 않은 폴더이면서 한 트리(폴더) 안에서 자식 노드(하위 항목)를 가지고 있고 자신과 동등한 노드(동등한 위치)이어야 한다. 실행 화면은 아래와 같다.

Name	Size	Kind	Date Modified
/	--	Drive	13. 3. 30. 오전 10:15
Applications	--	Folder	14. 1. 10. 오후 4:45
Developer	--	Folder	14. 1. 9. 오전 9:44
Applications	--	Folder	12. 11. 26. 오전 10:03
Documentation	--	Folder	13. 8. 14. 오전 10:15
Examples	--	Folder	12. 11. 26. 오전 9:50
Project	--	Folder	14. 1. 9. 오전 9:53
Tools	--	Folder	14. 1. 2. 오후 10:30
Library	--	Folder	14. 1. 9. 오전 9:45
System	--	Folder	13. 3. 30. 오후 3:02
Users	--	Folder	12. 3. 9. 오후 6:04
사용 설명서와 정보	--	Folder	12. 3. 9. 오후 6:04

위의 실행 화면을 본 결과 Application이라는 폴더를 열지 않았으므로 가지 표시가 나타나고 Developer라는 폴더를 열었으므로 가지 표시가 나타나지 않는 것을 알 수 있다.

- `QTreeView::branch:has-children:!has-siblings:closed {background: gray;}` : has-children 이므로 한 트리(폴더)안에서 자식 노드(하위 항목)를 가지고 있고 !has-

`siblings` 이므로 자신과 동등한 노드(동등한 위치)를 가지지 않는 열지 않은 폴더이어야 한다. 배경색상은 회색이다. 따라서 실행화면은 아래와 같다.

Name	Size	Kind	Date Modified
/	--	Drive	13. 3. 30. 오전 10:15
Applications	--	Folder	14. 1. 10. 오후 4:45
Developer	--	Folder	14. 1. 9. 오전 9:44
Applications	--	Folder	12. 11. 26. 오전 10:03
Documentation	--	Folder	13. 8. 14. 오전 10:15
Examples	--	Folder	12. 11. 26. 오전 9:50
Project	--	Folder	14. 1. 9. 오전 9:53
Tools	--	Folder	14. 1. 2. 오후 10:30
Library	--	Folder	14. 1. 9. 오전 9:45
System	--	Folder	13. 3. 30. 오후 3:02
Users	--	Folder	12. 3. 9. 오후 6:04
사용 설명서와 정보	--	Folder	12. 3. 9. 오후 6:04

위의 실행 화면 결과 'Tools'폴더와 '사용 설명서와 정보'폴더는 한 트리(폴더) 안에서 `!has-siblings` 이므로 다음과 같이 표현되고 있다.

- QTreeView::branch::open:has-children:has-siblings {background: magenta;} : open 은 열린 폴더를 의미하고 has-children 이므로 한 트리(폴더)안에서 자식 노드(하위 항목)를 가지고 있고 has-siblings 이므로 자신과 동등한 노드(동등한 위치)를 가지고 있는 폴더여야 한다. 즉, open 은 closed 와 반대 개념이라고 생각하면 된다. 배경색상은 자홍색이고 실행화면은 아래와 같다.

Name	Size	Kind	Date Modified
/	--	Drive	14. 1. 10. 오후 4:45
Applications	--	Folder	12. 3. 30. 오전 10:31
ALZip.app	--	Folder	12. 10. 29. 오전 9:24
Address Book.app	--	Folder	12. 10. 29. 오전 9:08
Contents	--	Folder	12. 10. 29. 오전 9:24
App Store.app	--	Folder	13. 7. 3. 오전 9:30
Automator.app	--	Folder	12. 10. 29. 오전 9:24
Calculator.app	--	Folder	12. 10. 29. 오전 9:24
Chess.app	--	Folder	12. 10. 29. 오전 9:24
Cornerstone.app	--	Folder	13. 3. 30. 오전 10:52
DVD Player.app	--	Folder	12. 10. 29. 오전 9:24
Dashboard.app	--	Folder	12. 10. 29. 오전 9:24
Dictionary.app	--	Folder	12. 10. 29. 오전 9:24
Evernote.app	--	Folder	13. 5. 24. 오전 10:44
FaceTime.app	--	Folder	12. 10. 29. 오전 9:24
Firefox.app	--	Folder	13. 12. 6. 오전 6:51
Font Book.app	--	Folder	12. 10. 29. 오전 9:24
GarageBand.app	--	Folder	12. 3. 20. 오후 2:30
Google Chrome.app	--	Folder	14. 1. 14. 오전 12:38
HanwordViewer.app	--	Folder	13. 5. 20. 오후 1:39
Image Capture.app	--	Folder	12. 10. 29. 오전 9:24
Keynote.app	--	Folder	13. 3. 4. 오후 4:45

'Applications'폴더와 'Address Book.app'폴더는 현재 열린 폴더이고 자식 노드(하위 항목)을 가지고 있고 자신과 동등한 노드(동등한 위치)를 가지고 있는 폴더이므로 위의 실행화면과 같은 결과가 나온다.

Name	Size	Kind	Date Modified
/	--	Drive	13. 3. 30. 오전 10:15
Applications	--	Folder	14. 1. 10. 오후 4:45
Developer	--	Folder	14. 1. 9. 오전 9:44
Library	--	Folder	14. 1. 9. 오전 9:45
System	--	Folder	13. 3. 30. 오후 3:02
Users	--	Folder	12. 3. 9. 오후 6:04
Shared	--	Folder	12. 3. 9. 오후 6:08
byeongcheongwon	--	Folder	14. 1. 22. 오전 9:59
사용 설명서와 정보	687 KB	pdf File	11. 6. 15. 오전 7:10
AirPort Extreme and Bluetooth reg...	73 KB	pdf File	08. 4. 8. 오전 6:11
ENERGY STAR.pdf	3.0 MB	pdf File	11. 6. 16. 오전 3:31
Mac mini Users Guide.pdf	14 KB	rtf File	08. 9. 20. 오전 3:35

'Users' 폴더와 '사용 설명서와 정보' 폴더는 둘 다 현재 폴더가 열려 있고, 자식 노드(하위 항목)를 가짐에도 불구하고 '사용 설명서와 정보' 폴더는 분홍색 표시가 되어 있지 않다. 그 이유는 '사용 설명서와 정보' 폴더가 **!has-siblings** 이기 때문이다.

- **QTreeView::branch::open::has-children:!has-siblings {background: green;}** : 현재 **open** 이므로 열려 있는 폴더이고 한 트리(폴더) 안에서 자식 노드(하위 항목)을 가지고 있고 **!has-siblings** 이므로 즉, 주변에 동등한 노드(동등한 위치)를 가지고 있지 않은 폴더를 표시해 주는 것이다. 배경색상은 초록색이고 실행화면은 아래와 같다.

Name	Size	Kind	Date Modified
/	--	Drive	13. 3. 30. 오전 10:15
Applications	--	Folder	14. 1. 10. 오후 4:45
Developer	--	Folder	14. 1. 9. 오전 9:44
Applications	--	Folder	12. 11. 26. 오전 10:03
Documentation	--	Folder	13. 8. 14. 오전 10:15
Examples	--	Folder	12. 11. 26. 오전 9:50
Project	--	Folder	14. 1. 9. 오전 9:53
Tools	--	Folder	14. 1. 2. 오후 10:30
Library	--	Folder	14. 1. 9. 오전 9:45
System	--	Folder	13. 3. 30. 오후 3:02
Users	--	Folder	12. 3. 9. 오후 6:04
사용 설명서와 정보	687 KB	pdf File	11. 6. 15. 오전 7:10
AirPort Extreme and Bluetoo...	73 KB	pdf File	08. 4. 8. 오전 6:11
ENERGY STAR.pdf	3.0 MB	pdf File	11. 6. 16. 오전 3:31
Mac mini Users Guide.pdf	14 KB	rtf File	08. 9. 20. 오전 3:35

'Developer' 폴더와 '사용 설명서와 정보' 폴더 둘 다 열린 폴더이고 자식 노드(하위 항목)를 가짐에도 불구하고 '사용 설명서와 정보' 폴더에 표시가 나타난 이유는 이 폴더가 **!has-siblings** 폴더이기 때문이다.

- **QTreeView::branch::has-siblings:!adjoins-item {border-image:url(:/new/prefix1stylesheet-vline.png);}** : 하위 항목을 가리키는 가지의 디자인 모양을 보여주고 있다. 먼저 **has-siblings** 이란 한 트리(폴더) 안에 동등한 노드(동등한 위치)에 있는 폴더를 의미하고 **!adjoins-item** 은 인접하지 않은 아이템들을 뜻하는 것으로써, 동등한 위치에 있는 폴더이면서 서로 인접하고 있지 않는 아이템들을 묶고 있다. 여기서 '**!**'은 **not**(부정)을 의미한다. 이 가지의 모양은 위의 경로의 이미지와 같으므로 명령을 실행하면 아래의 화면과 같다.

Name	Size	Kind	Date Modified
/		-- Drive	13. 3. 30. 오전 10:15
Applications		-- Folder	14. 1. 10. 오후 4:45
ALZip.app		-- Folder	12. 3. 30. 오전 10:31
Contents		-- Folder	12. 3. 30. 오전 10:31
CodeResources	7 KB	File	12. 3. 30. 오전 10:30
Info.plist	6 KB	plist File	12. 3. 30. 오전 10:30
MacOS		-- Folder	12. 1. 26. 오후 7:58
PkgInfo	8 byte(s)	File	12. 3. 30. 오전 10:30
Resources		-- Folder	11. 8. 16. 오후 7:07
_CodeSignature		-- Folder	11. 8. 16. 오후 7:07
_MASReceipt		-- Folder	12. 3. 30. 오전 10:31
Address Book.app		-- Folder	12. 10. 29. 오전 9:24
App Store.app		-- Folder	13. 7. 3. 오전 9:30
Automator.app		-- Folder	12. 10. 29. 오전 9:24
Calculator.app		-- Folder	12. 10. 29. 오전 9:24
Chess.app		-- Folder	12. 10. 29. 오전 9:24
Cornerstone.app		-- Folder	13. 3. 30. 오전 10:52
DVD Player.app		-- Folder	12. 10. 29. 오전 9:24
Dashboard.app		-- Folder	12. 10. 29. 오전 9:24
Dictionary.app		-- Folder	12. 10. 29. 오전 9:24
Evernote.app		-- Folder	13. 5. 24. 오전 10:44
FaceTime.app		-- Folder	12. 10. 29. 오전 9:24
Firefox.app		-- Folder	13. 12. 6. 오전 6:51
Font Book.app		-- Folder	12. 10. 29. 오전 9:24
GarageBand.app		-- Folder	12. 3. 20. 오후 2:30
Google Chrome.app		-- Folder	14. 1. 14. 오전 12:38
HanwordViewer.app		-- Folder	13. 5. 20. 오후 1:39
Image Capture.app		-- Folder	12. 10. 29. 오전 9:24
Keynote.app		-- Folder	13. 3. 4. 오후 4:45

Name	Size	Kind	Date Modified
Photo Booth.app		-- Folder	12. 10. 29. 오전 9:24
Preview.app		-- Folder	12. 10. 29. 오전 9:24
Producteev.app		-- Folder	13. 5. 24. 오전 11:02
Python 2.7		-- Folder	13. 5. 20. 오후 1:13
QuickTime Player.app		-- Folder	13. 7. 3. 오전 9:30
SCToolBarButton.app		-- Folder	13. 3. 30. 오전 10:44
SQLiteManager.app		-- Folder	13. 7. 29. 오전 12:57
Safari.app		-- Folder	13. 7. 3. 오전 9:30
SketchBookExpress.app		-- Folder	13. 5. 23. 오후 3:52
Stickies.app		-- Folder	12. 10. 29. 오전 9:24
System Preferences.app		-- Folder	12. 10. 29. 오전 9:24
TextEdit.app		-- Folder	12. 10. 29. 오전 9:24
Time Machine.app		-- Folder	12. 10. 29. 오전 9:24
Utilities		-- Folder	14. 1. 10. 오전 10:38
Wacom Tablet.localized		-- Folder	13. 2. 6. 오전 6:46
Xcode.app		-- Folder	13. 2. 7. 오전 8:09
iBooks Author.app		-- Folder	13. 5. 24. 오전 11:27
iCal.app		-- Folder	12. 10. 29. 오전 9:24
iChat.app		-- Folder	12. 10. 29. 오전 9:24
iMovie.app		-- Folder	11. 10. 11. 오전 9:33
iPhoto.app		-- Folder	12. 3. 20. 오후 2:37
iTunes.app		-- Folder	13. 7. 3. 오전 9:30
Developer		-- Folder	14. 1. 9. 오전 9:44
Library		-- Folder	14. 1. 9. 오전 9:45
System		-- Folder	13. 3. 30. 오후 3:02
Users		-- Folder	12. 3. 9. 오후 6:04
사용 설명서와 정보		-- Folder	12. 3. 9. 오후 6:04
Airport Extreme and Bluetoo...	687 KB	pdf File	11. 6. 15. 오전 7:10
ENERGY STAR.pdf	73 KB	pdf File	08. 4. 8. 오전 6:11
Mac mini Users Guide.pdf	3.0 MB	pdf File	11. 6. 16. 오전 3:31
The FreeType Project Licens...	14 KB	rtf File	08. 9. 20. 오전 3:35

여기서 '사용 설명서와 정보' 폴더는 !has-siblings 이므로 표시가 되지 않는다.

- QTreeView::branch:has-siblings:adjoins-item {border-image: url(:/new/prefix1/stylesheets/branch-more.png);} : has-siblings 이므로 한 트리(폴더) 안에 동등한 노드(동등한 위치)이면서 adjoins-item 이므로 인접한 폴더만 가지 표시를 해주게 된다. 이 가지의 모양은 위의 경로의 이미지와 같으므로 명령을 실행하면 아래의 화면과 같다.

Name	Size	Kind	Date Modified
/		Drive	13. 3. 30. 오전 10:15
Applications		Folder	14. 1. 10. 오후 4:45
ALZip.app		Folder	12. 3. 30. 오전 10:31
Contents		Folder	12. 3. 30. 오전 10:31
CodeResources	7 KB	File	12. 3. 30. 오전 10:30
Info.plist	6 KB	plist File	12. 3. 30. 오전 10:30
MacOS		Folder	12. 1. 26. 오후 7:58
PkgInfo		File	12. 3. 30. 오전 10:30
Resources		Folder	11. 8. 16. 오후 7:07
_CodeSignature		Folder	11. 8. 16. 오후 7:07
_MASReceipt		Folder	12. 3. 30. 오전 10:31
Address Book.app		Folder	12. 10. 29. 오전 9:24
App Store.app		Folder	13. 7. 3. 오전 9:30
Automator.app		Folder	12. 10. 29. 오전 9:24
Calculator.app		Folder	12. 10. 29. 오전 9:24
Chess.app		Folder	12. 10. 29. 오전 9:24
Cornerstone.app		Folder	13. 3. 30. 오전 10:52
DVD Player.app		Folder	12. 10. 29. 오전 9:24
Dashboard.app		Folder	12. 10. 29. 오전 9:24
Dictionary.app		Folder	12. 10. 29. 오전 9:24
Evernote.app		Folder	13. 5. 24. 오전 10:44
FaceTime.app		Folder	12. 10. 29. 오전 9:24
Firefox.app		Folder	13. 12. 6. 오전 6:51
Font Book.app		Folder	12. 10. 29. 오전 9:24
GarageBand.app		Folder	12. 3. 20. 오후 2:30
Google Chrome.app		Folder	14. 1. 14. 오전 12:38
HanwordViewer.app		Folder	13. 5. 20. 오후 1:39
Image Capture.app		Folder	12. 10. 29. 오전 9:24

- QTreeView::branch:!has-children:!has-siblings:adjoins-item {border-image:

url(:/new/prefix1/stylesheets/branch-end.png);} : !has-children 이므로 자식을 가지지

않는(하위항목을 가지지 않는) 폴더이고 동시에 자신과 동등한 노드(동등한 위치)를 갖지도 않아야 하면서 자신과 인접한 아이템들을 표시해야 한다. 이 가지의 모양은 위의 경로의 이미지와 같으므로 명령을 실행하면 아래의 화면과 같다.

Name	Size	Kind	Date Modified
/		Drive	13. 3. 30. 오전 10:15
Applications		Folder	14. 1. 10. 오후 4:45
ALZip.app		Folder	12. 3. 30. 오전 10:31
Contents		Folder	12. 3. 30. 오전 10:31
CodeResources	7 KB	File	12. 3. 30. 오전 10:30
Info.plist	6 KB	plist File	12. 3. 30. 오전 10:30
MacOS		Folder	12. 1. 26. 오후 7:58
ALZip	1.2 MB	File	12. 3. 30. 오전 10:30
bin		Folder	11. 8. 16. 오후 7:07
7z.so	2.2 MB	so File	12. 3. 30. 오전 10:30
PkgInfo		File	12. 3. 30. 오전 10:30
Resources		Folder	11. 8. 16. 오후 7:07
_CodeSignature		Folder	11. 8. 16. 오후 7:07
_MASReceipt		Folder	12. 3. 30. 오전 10:31
Address Book.app		Folder	12. 10. 29. 오전 9:24
App Store.app		Folder	13. 7. 3. 오전 9:30
Automator.app		Folder	12. 10. 29. 오전 9:24
Calculator.app		Folder	12. 10. 29. 오전 9:24
Chess.app		Folder	12. 10. 29. 오전 9:24
Cornerstone.app		Folder	13. 3. 30. 오전 10:52
DVD Player.app		Folder	12. 10. 29. 오전 9:24
Dashboard.app		Folder	12. 10. 29. 오전 9:24
Dictionary.app		Folder	12. 10. 29. 오전 9:24

현재 '7z.so' 파일을 보면 'branch-end' 이미지가 들어가 있다.

- QTreeView::branch:has-children:!has-siblings:closed,

QTreeView::branch:closed:has-children:has-siblings {border-image: none;

image: url(:/new/prefix1/stylesheets/branch-closed.png);} : 한 트리(폴더)안에서 자식(하위)

목록)을 가지고 있고 동등한 노드(동등한 위치)를 포함하지 않고, 열려 있지 않은 폴더일 때와 한 트리(폴더)안에서 열려 있지 않고 자식(하위 목록)을 가지고 있고 동등한 노드(동등한 위치) 두 가지 다 포함하고 있을 경우에 다음의 경로에 따른 이미지를 삽입할 수 있다. 실행화면 결과는 아래와 같다.

Name	Size	Kind	Di
/	--	Drive	1
Applications	--	Folder	1
ALZip.app	--	Folder	1
Address Book.app	--	Folder	1
Contents	--	Folder	1
App Store.app	--	Folder	1
Automator.app	--	Folder	1
Calculator.app	--	Folder	1
Chess.app	--	Folder	1
Cornerstone.app	--	Folder	1
DVD Player.app	--	Folder	1
Dashboard.app	--	Folder	1
Dictionary.app	--	Folder	1
Evernote.app	--	Folder	1
FaceTime.app	--	Folder	1
Firefox.app	--	Folder	1
Font Book.app	--	Folder	1
GarageBand.app	--	Folder	1
Google Chrome.app	--	Folder	1

'Application' 폴더와 'Address Book.app' 폴더는 열려 있으므로 화살표 표시가 없어진다.

- QTreeView::branch::open:has-children:!has-siblings, QTreeView::branch::open:has-children:has-siblings {border-image: none; image:url(:/new/prefix1/stylesheets-branch-open.png);} : 항상 열려있는 폴더이고 한 트리(폴더) 안에서 자식 노드(하위 목록)을 가지고 있고, 동등한 노드(동등한 위치)를 포함하고 있지 않은 폴더와 항상 열려있는 폴더이고 한 트리(폴더)안에서 자식 노드(하위 목록)을 가지고 있고, 동등한 노드(동등한 위치)를 포함하고 있는 폴더가 있을 때 두 가지 경우 모두 다 위의 경로와 같은 이미지를 삽입할 수 있다. 실행화면은 아래와 같다.

Name	Size	Kind	Date Modified
/	--	Drive	13. 3. 30. 오전 10:15
Applications	--	Folder	14. 1. 10. 오후 4:45
Developer	--	Folder	14. 1. 9. 오전 9:44
Applications	--	Folder	12. 11. 26. 오전 10:03
Documentation	--	Folder	13. 8. 14. 오전 10:15
Qt	--	Folder	12. 11. 26. 오전 9:50
Examples	--	Folder	12. 11. 26. 오전 9:50
Project	--	Folder	14. 1. 9. 오전 9:53
Tools	--	Folder	14. 1. 2. 오후 10:30
Library	--	Folder	14. 1. 9. 오전 9:45
System	--	Folder	13. 3. 30. 오후 3:02
Users	--	Folder	12. 3. 9. 오후 6:04
사용 설명서와 정보	--	Folder	12. 3. 9. 오후 6:04

위에서 branch 를 다루었던 것들을 합쳐 본 결과의 실행화면은 아래와 같다.

Name	Size	Kind	Date Modified
▼ /		-- Drive	13. 3. 30. 오전 10:15
▼ Applications		-- Folder	14. 1. 10. 오후 4:45
▶ ALZip.app		-- Folder	12. 3. 30. 오전 10:31
▼ Address Book.app		-- Folder	12. 10. 29. 오전 9:24
▼ Contents		-- Folder	12. 10. 29. 오전 9:08
Info.plist	6 KB	plist File	12. 10. 29. 오전 9:04
▼ MacOS		-- Folder	12. 10. 29. 오전 9:08
Address Book	721 KB	File	12. 10. 29. 오전 9:04
PkgInfo	8 byte(s)	File	11. 10. 11. 오전 9:19
▼ Resources		-- Folder	12. 10. 29. 오전 9:22
▼ _CodeSignature		-- Folder	12. 10. 29. 오전 9:08
version.plist	462 byte(s)	plist File	12. 10. 29. 오전 9:04
▶ App Store.app		-- Folder	13. 7. 3. 오전 9:30
▶ Automator.app		-- Folder	12. 10. 29. 오전 9:24
▶ Calculator.app		-- Folder	12. 10. 29. 오전 9:24
▶ Chess.app		-- Folder	12. 10. 29. 오전 9:24
▶ Cornerstone.app		-- Folder	13. 3. 30. 오전 10:52
▶ DVD Player.app		-- Folder	12. 10. 29. 오전 9:24
▶ Dashboard.app		-- Folder	12. 10. 29. 오전 9:24
▶ Dictionary.app		-- Folder	12. 10. 29. 오전 9:24
▶ Evernote.app		-- Folder	13. 5. 24. 오전 10:44
▶ FaceTime.app		-- Folder	12. 10. 29. 오전 9:24
▶ Firefox.app		-- Folder	13. 12. 6. 오전 6:51