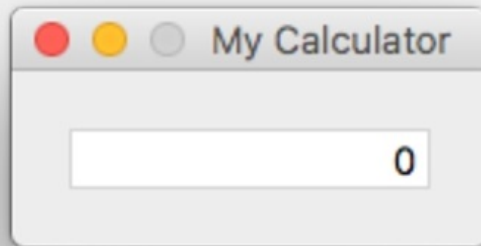


계산기 프로젝트



- 계산기가 가질 기능
 - 숫자 키패드: 입력
 - 사칙연산
 - 괄호
 - 지우기
 - 상수 입력
 - 함수 계산

우선은 숫자 표시 창부터



```
from PyQt5.QtCore import Qt
from PyQt5.QtWidgets ...
from PyQt5.QtWidgets ...

class Calculator(QWidget):
    ...

if __name__ == '__main__':
    import sys
    app = QApplication(sys.argv)
    calc = Calculator()
    calc.show()
    sys.exit(app.exec_())
```

```
class Calculator(QWidget):
```

```
    def __init__(self, parent=None):
```

```
        super().__init__(parent)
```

mycalc1.py

```
        # Display Window
```

```
        self.display = QLineEdit('0')
```

```
        self.display.setReadOnly(True)
```

```
        self.display.setAlignment(Qt.AlignRight)
```

```
        self.display.setMaxLength(15)
```

```
        # Layout
```

```
        mainLayout = QGridLayout()
```

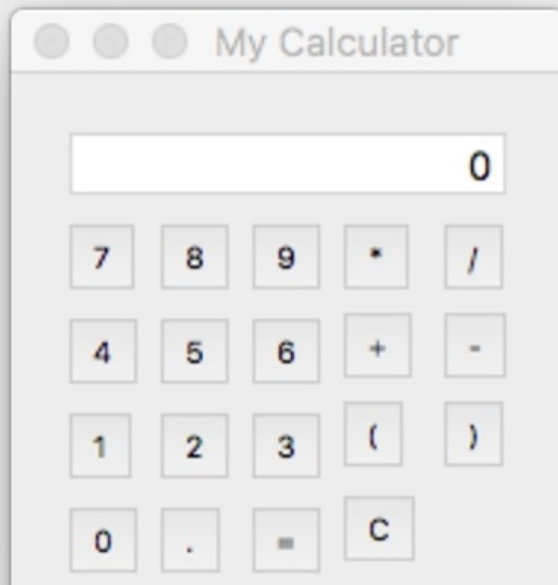
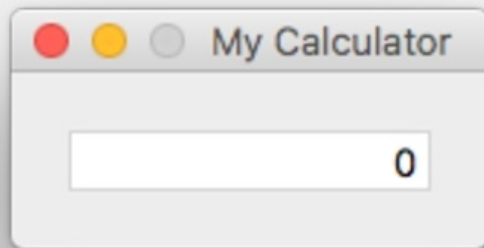
```
        mainLayout.setSizeConstraint(QLayout.SetFixedSize)
```

```
        mainLayout.addWidget(self.display, 0, 0, 1, 1)
```

```
        self.setLayout(mainLayout)
```

```
        self.setWindowTitle("My Calculator")
```

QGridLayout



```
mainLayout = QGridLayout()  
mainLayout.setSizeConstraint(QLayout.SetFixedSize)  
mainLayout.addWidget(self.display, 0, 0, 1, 1)
```

- `self.display` → QLineEdit 객체 (widget)
- `addWidget(..., 0, 0, 1, 1)`
 - 좌표 (0, 0) 에 배치
 - 가로로 1 그리드 차지
 - 세로로 1 그리드 차지
- 왼쪽과 같이 하려면, 가로로 몇 그리드를 차지하도록 해야?

숫자 키패드 버튼의 생성

- 버튼 생성 코드
 - 동일 (유사) 한 코드의 반복으로 이루어져 있음 → 나중에 개선

```
# Digit Buttons
```

```
self.digitButton = [x for x in range(0, 10)] → 이 행은 왜 있어야?
```

```
self.digitButton[0] = QPushButton()
```

```
self.digitButton[0].setText('0')
```

```
self.digitButton[1] = QPushButton()
```

```
self.digitButton[1].setText('1')
```

```
self.digitButton[2] = QPushButton()
```

```
self.digitButton[2].setText('2')
```

```
·  
·  
·
```

```
mycalc2.py
```

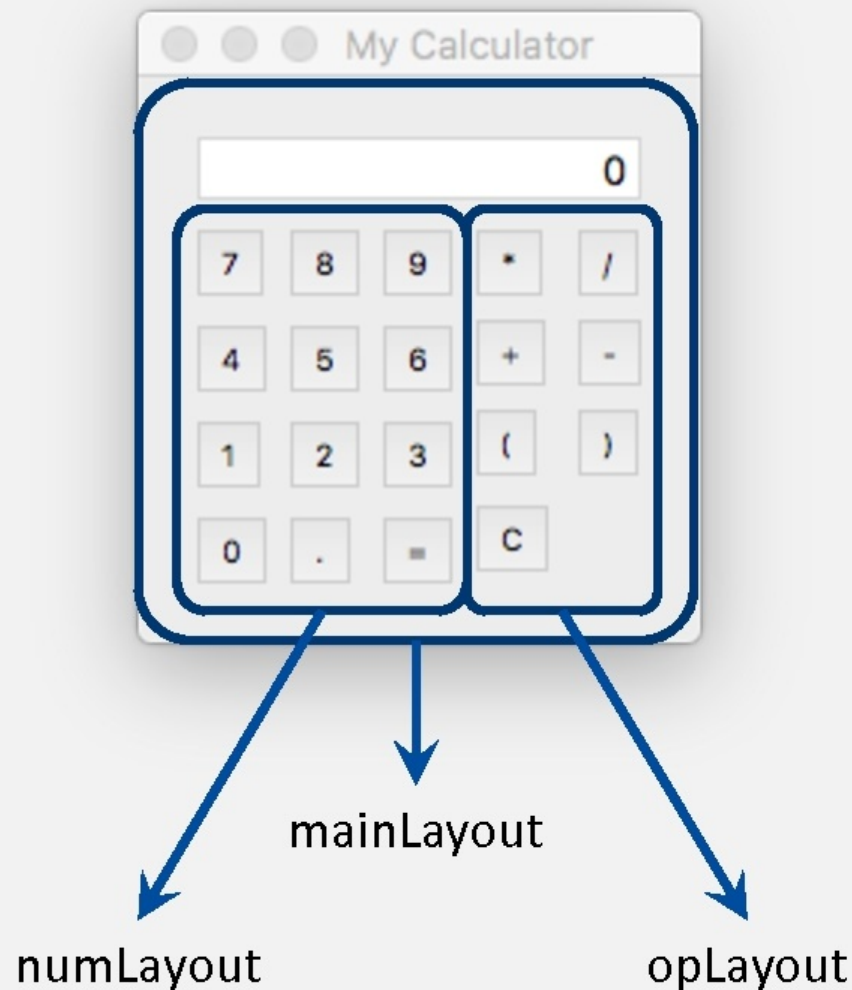
```
>>> v = [x for x in range(0, 10)]
```

```
>>> v
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
>>>
```

숫자, 연산기호의 버튼 배치



- 두 개의 sub layout 을 생성
 - 숫자 키패드: numLayout
 - [.] 버튼과 [=] 버튼은 별도로
 - 연산자 키패드: opLayout
- 이들을 mainLayout 에 배치

```
numLayout.addWidget(...)  
opLayout.addWidget(...)
```

```
mainLayout.addLayout(numLayout, ...)  
mainLayout.addLayout(opLayout, ...)
```


숫자 키패드 버튼의 배치

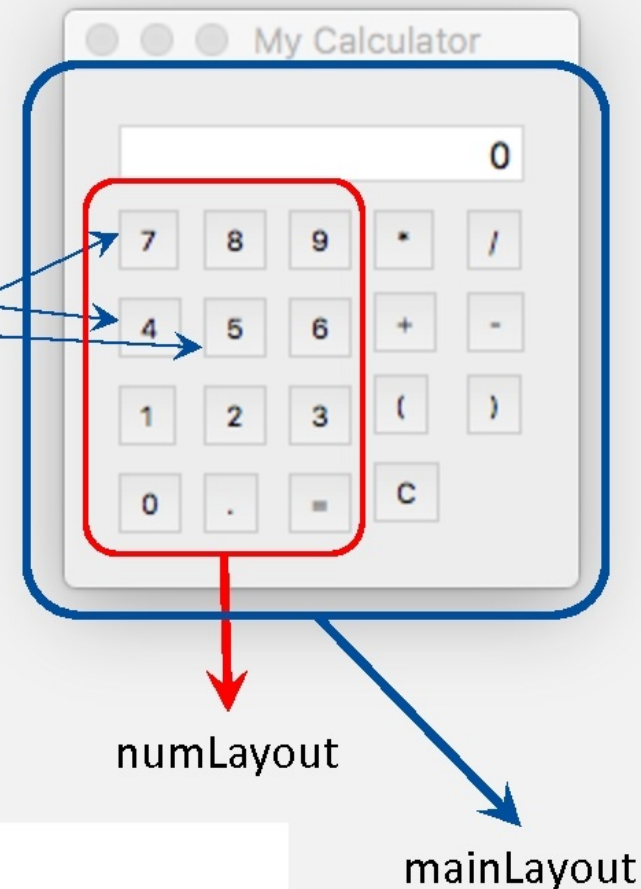
- 이것도 역시 동일 (유사) 한 코드의 반복
 - 마찬가지로 나중에 개선하기로 함

```
numLayout = QGridLayout()
```

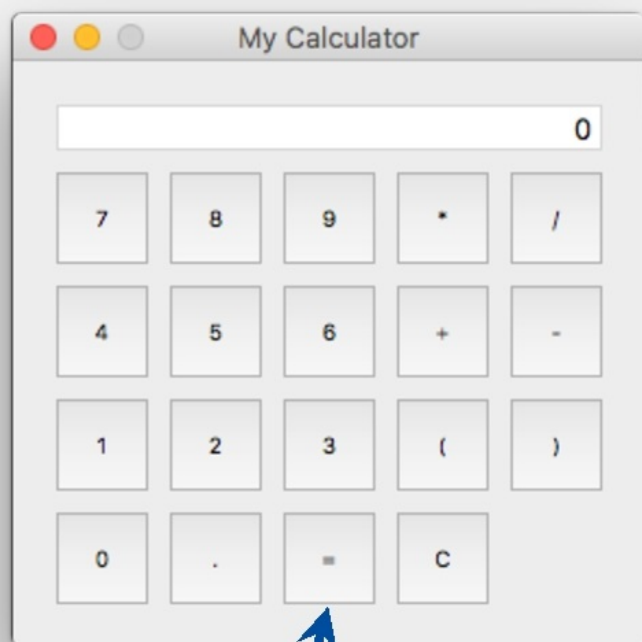
```
numLayout.addWidget(self.digitButton[0], 3, 0)
numLayout.addWidget(self.digitButton[1], 2, 0)
numLayout.addWidget(self.digitButton[2], 2, 1)
numLayout.addWidget(self.digitButton[3], 2, 2)
numLayout.addWidget(self.digitButton[4], 1, 0)
numLayout.addWidget(self.digitButton[5], 1, 1)
numLayout.addWidget(self.digitButton[6], 1, 2)
numLayout.addWidget(self.digitButton[7], 0, 0)
numLayout.addWidget(self.digitButton[8], 0, 1)
numLayout.addWidget(self.digitButton[9], 0, 2)
```

```
numLayout.addWidget(self.decButton, 3, 1)
numLayout.addWidget(self.eqButton, 3, 2)
```

```
mainLayout.addLayout(numLayout, 1, 0)
```



Button 클래스를 정의, 이용



버튼들의 크기가
일률적으로 달라졌는데?

- 새로운 클래스를 정의함으로써
 - 코드 반복을 줄일 수 있음
 - 공통의 속성을 손쉽게 다룰 수 있음

```
class Button(QToolButton):
```

mycalc3.py

```
def __init__(self, text):  
    super().__init__()   
    self.setSizePolicy(QSizePolicy.Expanding,  
                      QSizePolicy.Preferred)  
    self.setText(text)  
  
def sizeHint(self):  
    size = super(Button, self).sizeHint()  
    size.setHeight(size.height() + 20)  
    size.setWidth(max(size.width(), size.height()))  
    return size
```

새로 만든 클래스를 이용한 버튼의 생성

- 코드의 어떤 점이 개선되었나요?
 - 더 개선할 여지는 없나요? → 과제

Digit Buttons

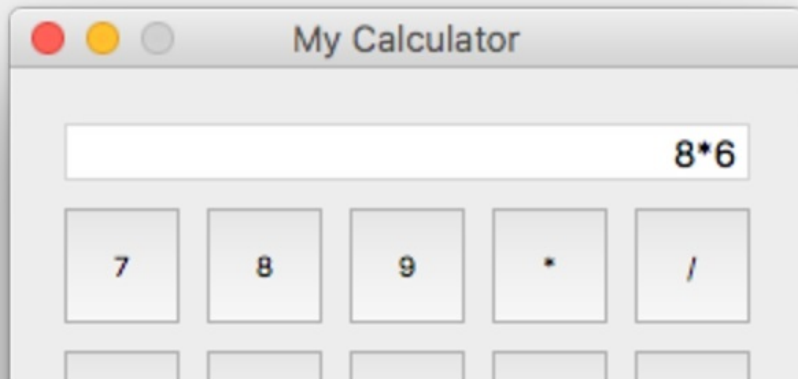
```
self.digitButton = [x for x in range(0, 10)]
```

```
self.digitButton[0] = Button('0')  
self.digitButton[1] = Button('1')  
self.digitButton[2] = Button('2')  
self.digitButton[3] = Button('3')  
self.digitButton[4] = Button('4')  
self.digitButton[5] = Button('5')  
self.digitButton[6] = Button('6')  
self.digitButton[7] = Button('7')  
self.digitButton[8] = Button('8')  
self.digitButton[9] = Button('9')
```

버튼 하나당 두 줄씩 필요했던 코드가
한 줄씩으로 줄었음

- 과연, 그것 뿐인가?
 - 유지보수 가능성
 - 재사용성
 - 가독성

연산 기능의 구현



```
def buttonClicked(self):
    button = self.sender()
    key = button.text()
    if key == '=':
        result = str(eval(self.display.text()))
        self.display.setText(result)
    elif key == 'C':
        self.display.setText("")
    else:
        self.display.setText(self.display.text() + key)
```

- 버튼 기능 (callback 함수) 구현
 - 숫자 버튼이 눌릴 때
 - 해당 숫자를 덧붙여 표시
 - 연산 기호가 눌릴 때
 - 연산 기호를 표시
 - [=] 버튼이 눌릴 때
 - 수식을 계산하여 결과를 표시
 - [C] 버튼이 눌릴 때
 - 현재 표시된 내용을 모두 지움

계산기의 연산 기능

- eval() 함수 (Python built-in function)
 - 입력: Python string
 - 출력: 위 입력을 Python 표현식 (expression) 으로 간주한 연산 결과를 담은 Python 객체

```
Python 2.7.12 (default, Oct 11 2016, 05:20:59)
[GCC 4.2.1 Compatible Apple LLVM 8.0.0 (clang-800.0.38)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> eval('3+5')
8
>>> eval('8*7')
56
>>> eval
<built-in function eval>
>>>
```

버튼 콜백의 등록

- 콜백 함수도 버튼을 생성할 때 인자로 받아서 처리하자!
 - 왜? 모두 같은 콜백이 호출되도록 할 거 아닌가?
 - 다시 한번 생각해 보자, 코드의 재사용성


Button 클래스의 생성자 (constructor)

```
def __init__(self, text, callback):  
    super().__init__()  
    self.setSizePolicy(QSizePolicy.Expanding,  
                      QSizePolicy.Preferred)  
    self.setText(text)  
    self.clicked.connect(callback)
```

여기서의 self 는 누구인가요?



여기서의 self 는 누구인가요?

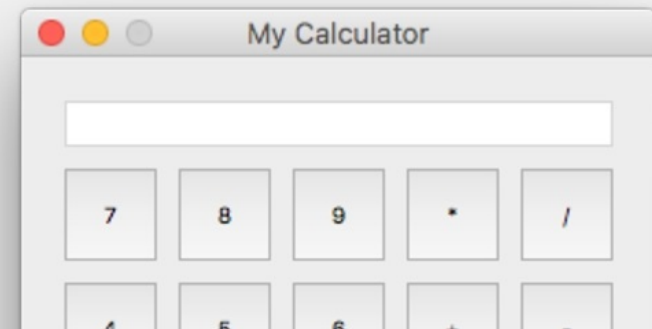
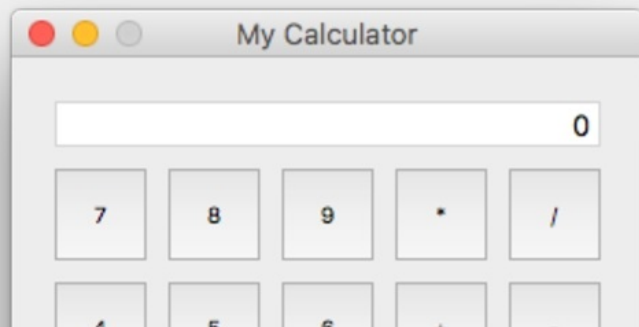


버튼 생성 코드

```
self.digitButton[0] = Button('0', self.buttonClicked)  
self.digitButton[1] = Button('1', self.buttonClicked)  
...
```

시작할 때의 표시창 내용을 변경

- '0' 이 표시되던 것을, 빈 문자열로 변경
 - 왜 변경했나요?



```
self.display = QLineEdit('0')
```

```
...
```

```
else:
```

```
    self.display.setText(self.display.text() + key)
```

```
self.display = QLineEdit()
```

```
>>> 04
```

```
4
```

```
>>> 073 → 이게 다 뭐지?
```

```
59
```

```
>>>
```

과제 - UI 구성 코드의 개선

1. 숫자 키패드 버튼 코드의 반복을 피할 것

버튼 UI 컴포넌트의 생성 코드
(숫자 키패드만)

```
# Digit Buttons  
self.digitButton = [x for x in range(0, 10)]
```

```
self.digitButton[0] = Button('0', self.buttonClicked)  
self.digitButton[1] = Button('1', self.buttonClicked)  
self.digitButton[2] = Button('2', self.buttonClicked)  
self.digitButton[3] = Button('3', self.buttonClicked)  
self.digitButton[4] = Button('4', self.buttonClicked)  
self.digitButton[5] = Button('5', self.buttonClicked)  
self.digitButton[6] = Button('6', self.buttonClicked)  
self.digitButton[7] = Button('7', self.buttonClicked)  
self.digitButton[8] = Button('8', self.buttonClicked)  
self.digitButton[9] = Button('9', self.buttonClicked)
```

동일 (유사) 코드의 반복이 심하다.

→ 결코 좋은 프로그래밍 방법이 아님

(버튼이 수천 개면 어떻게 할텐가?)

무슨 구조를 이용해서 프로그래밍 가능?

→ 순환문: for? while?

버튼 표면 텍스트는 어떻게 구성?

우선은 다른 버튼들은 그대로 두기로

과제 – UI 구성 코드의 개선

2. 버튼의 레이아웃 배치 코드도 개선

버튼 UI 컴포넌트의 레이아웃 배치 코드
(숫자 키패드만)

```
numLayout = QGridLayout()
```

```
numLayout.addWidget(self.digitButton[0], 3, 0)
numLayout.addWidget(self.digitButton[1], 2, 0)
numLayout.addWidget(self.digitButton[2], 2, 1)
numLayout.addWidget(self.digitButton[3], 2, 2)
numLayout.addWidget(self.digitButton[4], 1, 0)
numLayout.addWidget(self.digitButton[5], 1, 1)
numLayout.addWidget(self.digitButton[6], 1, 2)
numLayout.addWidget(self.digitButton[7], 0, 0)
numLayout.addWidget(self.digitButton[8], 0, 1)
numLayout.addWidget(self.digitButton[9], 0, 2)
```

동일 (유사) 코드의 반복이 심하다.

→ 결코 좋은 프로그래밍 방법이 아님

(버튼이 수천 개면 어떻게 할텐가?)

무슨 구조를 이용해서 프로그래밍 가능?

→ 순환문

버튼 배치 좌표에는, 물론, 규칙이 있다!

→ 어떤 규칙을 찾을 수 있는가?

숫자 키패드 버튼의 배치 (행, 열) 규칙

			행
			0
			1
			2
			3
열	0	1	2
	7	8	9
	4	5	6
	1	2	3
	0	.	=

- 행 번호의 규칙 (버튼이 나타내는 수에 대하여)
 - 수가 높으면 행 번호가 작다.
 - 한 행에 세 개씩의 수가 위치한다.
- 열 번호의 규칙 (버튼이 나타내는 수에 대하여)
 - 수가 하나 증가하면 열 번호도 하나 증가한다.
 - 하지만 열 번호가 3에 도달하지 않고 되돌아온다.

버튼 배치 (행, 열) 의 계산

			행
열	0	1	2
	7	8	9
	4	5	6
	1	2	3
	0	.	=

- 열 번호를 계산하려면?

i	$i \% 3$?
1	1	0
2	2	1
3	0	2
4	1	0
5	2	1
6	0	2
7	1	0
8	2	1
9	0	2

- 이 다음은 여러분의 노력으로

버튼 배치 (행, 열) 의 계산

- 행 번호를 계산하려면?

			행
7	8	9	0
4	5	6	1
1	2	3	2
0	.	=	3
열	0	1	2

i	$i / 3$
1	0
2	0
3	1
4	1
5	1
6	2
7	2
8	2
9	3

- 이 다음은 여러분의 노력으로

과제 결과물

- 버튼이 눌려지면 창에 입력되는 기능 구현
 - 숫자 버튼
 - 소수점 버튼
 - 괄호 버튼
 - 사칙연산 기호 버튼
- 연산 기능 실행 버튼
 - [=]
- 전체 지우기 버튼
 - [C]
- UI 구성 코드의 개선
 - 버튼 생성 코드 반복 줄이기
 - 버튼 배치 코드 반복 줄이기