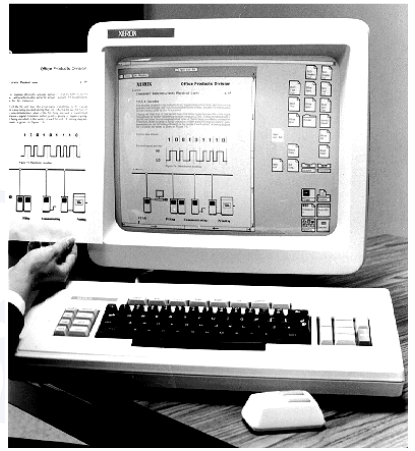
AiDA Lab.
Artificial Intelligence & Data Analysis Laboratory

GUI: Tkinter

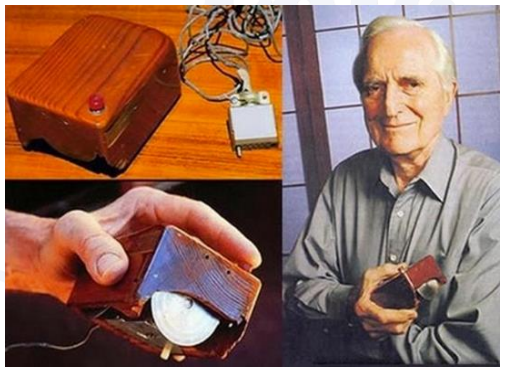
Seokhwan Yang

파이썬과 GUI

- **GUI (Graphical User Interface)**
  - **사용자가 그래픽을 통해 컴퓨터와 정보를 교환하는 작업 환경**
  - **GUI의 발달**
    - 최초의 GUI는 1958년 북미항공우주방위사령부의 SAGE 시스템
    - 민간 최초의 GUI는 1963년 MIT의 아이번 서덜랜드 교수의 스케치패드
    - 최초의 GUI 운영체제는 1973년 제록스 앨토(Alto) 컴퓨터
      - 흔히 Apple의 매킨토시라고 잘못 알려져 있음
  - **주로 마우스를 사용해서 입력을 처리함**



http://content.answers.com/main/content/img/CDE/_STAR1.GIF

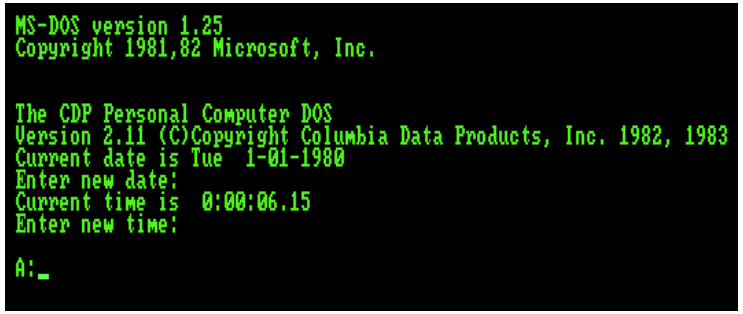


더글러스 엥겔비트가 개발한
최초의 마우스

https://i.imgur.com/a6u14O3.jpg

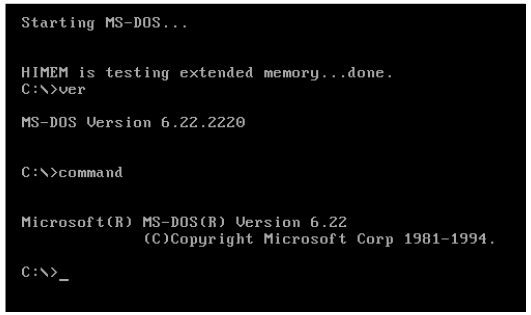
파이썬과 GUI

- **CUI (Character User Interface)**
  - **Command Line(명령줄)을 이용하여 컴퓨터와 정보를 교환하는 작업 환경**
  - **CLI(Command Line Interface) 또는 TUI(Text User Interface)라고도 부름**
  - **Terminal, Unix/Linux의 서버환경 등을 제외하면 이제는 찾아보기 어려움**

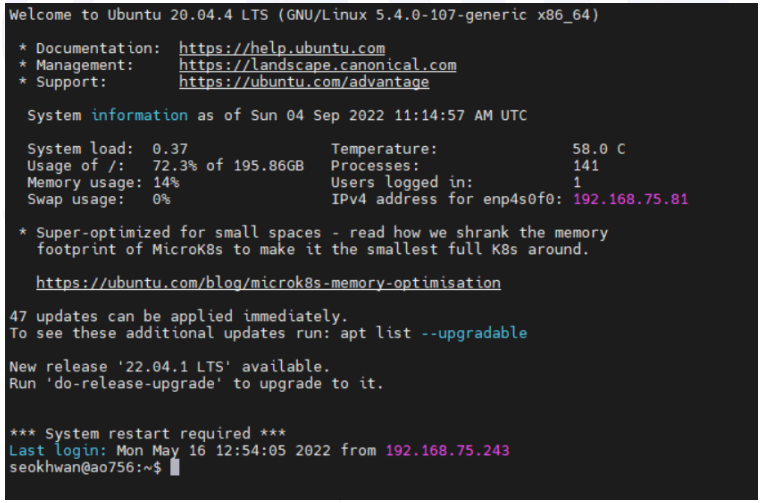


1980년 IBM에서 첫 PC를 개발하면서
도입한 MS-DOS 초기 버전

MSDOS 최초버전은 1.10(1982년 5월 발매)이며
IBM PC에 도입된 최초 버전은 1.25



MS Windows에 의존하지 않고 독립적으로
구동되는 마지막 MS DOS버전(6.22)



Ubuntu Server 화면

파이썬과 GUI

- **GUI를 개발하기 어려운 대표적인 언어**
  - **Java**
    - AWT, Swing 등의 라이브러리를 사용하여 GUI를 개발할 수 있으나 편의성, 성능이 떨어짐
    - Android의 경우에는 OS의 지원으로 인해 상당히 편리해짐
  - **Python**
    - Tkinter, PyQT, wxPython, PyGTK 등의 라이브러리를 사용하여 GUI를 개발할 수 있음
    - 역시 편의성과 성능이 떨어짐
    - 그러나 최근 Python의 인기가 높아짐에 따라 많이 개선되었음
  - **그 외 기타 언어들...**

• Java는 직접 GUI를 개발하기보다는 Web Browser의 지원에 힘입어 Web을 기반으로 개발하는 경우가 보편화되었으며 더 편리함

• 타 언어에 비해 Python으로 GUI를 개발하는 것이 더 쉽다고 주장하는 그룹도 있음
• 그러나 개인적인 생각으로는... 직접 파이썬으로 GUI를 개발해 본 결과에 기반하여... 동의하기 어려움

Tkinter

- **tkinter란?**
  - **파이썬에서 Tcl/Tk 툴킷을 사용하는데 필요한 인터페이스 모듈**
    - Tcl: 존 오스터하우트가 개발한 스크립트 언어 (Tcl: Tool Command Language)
      빠른 프로토타이핑, 스크립트 프로그램, GUI 및 테스팅에 많이 사용됨
    - Tk: Tcl을 위한 GUI 툴킷
  - **파이썬은 Tkinter 모듈을 기본적으로 내장하여 배포되므로 PyQt 등의 모듈 처럼 따로 설치할 필요가 없음**
  - **Google CoLab 환경에서는 사용불가능**

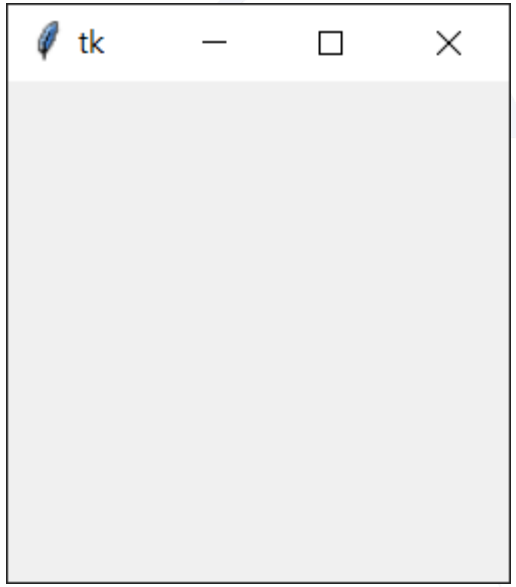
- GUI 기본 구성 및 동작

윈도우 표시하기

- 모듈 import
  - 파이썬 2.x에서는 Tkinter를 import하고 파이썬 3.x에서는 tkinter를 import
- Tk 클래스 객체(root) 생성 후 mainloop 실행

gui001.py

```
newvenv > gui001.py > ...
1   import tkinter
2
3   root = tkinter.Tk()
4   root.mainloop()
5
```



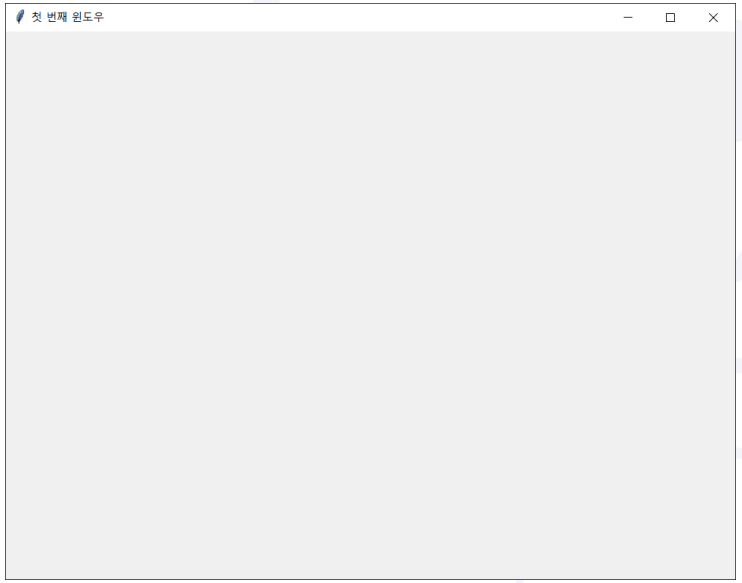
제목과 크기 지정하기

- 제목: title()
- 크기: geometry()
  - 800x600 과 같이 크기를 지정할 때, x는 소문자를 사용함

gui001.py × gui002.py ×

newvenv > gui002.py > ...

```
1  import tkinter
2
3  root = tkinter.Tk()
4
5  root.title("첫 번째 윈도우")
6  root.geometry("800x600")
7  root.mainloop()
8
```



PPT에 붙이기 위한 것이므로 크기의 표시는 어떻게 할 방법이 없음

라벨 배치하기

- 라벨: Label()
- 배치: place()

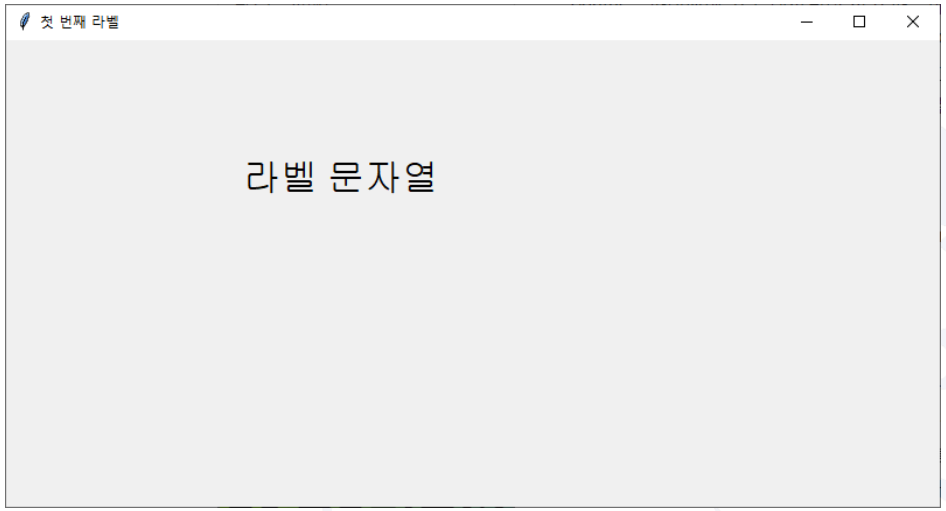
라벨 변수명=tkinter.Label(윈도우 객체, text="라벨 문자열", font=("폰트 명", 폰트 크기))

라벨 변수명.place(x=X 좌표, y=Y 좌표)

gui001.py | gui002.py | gui003.py

newvenv > gui003.py > ...

```
1  import tkinter
2
3  root = tkinter.Tk()
4
5  root.title("첫 번째 라벨")
6  root.geometry("800x400")
7  label = tkinter.Label(root, text="라벨 문자열", font=("System", 24))
8  label.place(x=200, y=100)
9  root.mainloop()
10
```



• 라벨, 버튼과 같은 GUI의 구성요소들을 일반적으로 위젯(Widget)이라고 부름

사용 가능한 폰트 확인하기

- **폰트 종류 확인: tkinter.font.families()**

gui002.py | gui003.py | gui004.py

newvenv > gui004.py > ...

```
1  import tkinter
2  import tkinter.font
3
4  root = tkinter.Tk()
5  print(tkinter.font.families())
6
```

사용자의 시스템에 따라 모두 다른 결과가 나옴

터미널 JUPYTER 디버그 콘솔 문제 출력 powershell

```
(newvenv) PS C:\Workspaces\Dev\Python\newvenv> python .\gui004.py
('System', '@System', 'Terminal', '@Terminal', 'Fixedsys', '@Fixedsys', 'Modern', 'Roman', 'Script', 'Courier', 'MS Serif', 'MS Sans Serif', 'Small Fonts', 'Marlett', 'Arial', 'Arabic Transparent', 'Arial Baltic', 'Arial CE', 'Arial CYR', 'Arial Greek', 'Arial TUR', 'Arial Black', 'Bahnschrift Light', 'Bahnschrift SemiLight', 'Bahnschrift', 'Bahnschrift SemiBold', 'Bahnschrift Light SemiCondensed', 'Bahnschrift SemiLight SemiConde', 'Bahnschrift SemiCondensed', 'Bahnschrift SemiBold SemiConden', 'Bahnschrift Light Condensed', 'Bahnschrift SemiLight Condensed', 'Bahnschrift Condensed', 'Bahnschrift SemiBold Condensed', 'Calibri', 'Calibri Light', 'Cambria', 'Cambria Math', 'Candara', 'Candara Light', 'Comic Sans MS', 'Consolas', 'Constantia', 'Corbel', 'Corbel Light', 'Courier New', 'Courier New Baltic', 'Courier New CE', 'Courier New CYR', 'Courier New Greek', 'Courier New TUR', 'Ebrima', 'Franklin Gothic Medium', 'Gabriola', 'Gadugi', 'Georgia', 'Impact', 'Ink Free', 'Javanese Text', 'Leelawadee UI', 'Leelawadee UI Semilight', 'Lucida Console', 'Lucida Sans Unicode', '맑은 고딕', '@맑은 고딕', '맑은 고딕 Semilight', '@맑은 고딕 Semilight', 'Microsoft Himalaya', 'Microsoft JhengHei', '@Microsoft JhengHei', 'Microsoft JhengHei UI', '@Microsoft JhengHei UI', 'Microsoft JhengHei Light', '@Microsoft JhengHei Light', 'Microsoft JhengHei UI Light', '@Microsoft JhengHei UI Light', 'Microsoft New Tai Lue', 'Microsoft PhagsPa', 'Microsoft Sans Serif', 'Microsoft Tai Le', 'Microsoft YaHei', '@Microsoft YaHei', 'Microsoft YaHei UI', '@Microsoft YaHei UI', 'Microsoft YaHei Light', '@Microsoft YaHei Light', 'Microsoft YaHei UI Light', '@Microsoft YaHei UI Light', 'Microsoft Yi Baiti', 'MingLiU-ExtB', '@MingLiU-ExtB', 'PMingLiU-ExtB', '@PMingLiU-ExtB', 'MingLiU_HKSCS-ExtB', '@MingLiU_HKSCS-ExtB', 'Mongolian Baiti', 'MS Gothic', '@MS Gothic', 'MS UI Gothic', '@MS UI Gothic', 'MS PGothic', '@MS PGothic', 'MV Boli', 'Myanmar Text', 'Nirmala UI', 'Nirmala UI Semilight', 'Palatino Linotype', 'Segoe MDL2 Assets', 'Segoe Print', 'Segoe Script', 'Segoe UI', 'Segoe UI Black', 'Segoe UI Emoji', 'Segoe UI Historic', 'Segoe UI Light', 'Segoe UI Semibold', 'Segoe UI Semilight', 'Segoe UI Symbol', 'SimSun', '@SimSun', 'NSimSun', '@NSimSun', 'SimSun-ExtB', '@SimSun-ExtB', 'Sitka Small', 'Sitka Text', 'Sitka Subheading', 'Sitka Heading', 'Sitka Display', 'Sitka Banner', 'Sylfaen', 'Symbol', 'Tahoma', 'Times New Roman', 'Times New Roman Baltic', 'Times New Roman CE', 'Times New Roman CYR', 'Times New Roman Greek', 'Times New Roman TUR', 'Trebuchet MS', 'Verdana', 'Webdings', 'Wingdings', 'Yu Gothic', '@Yu Gothic', 'Yu Gothic UI', '@Yu Gothic UI', 'Yu Gothic UI Semibold', '@Yu Gothic UI Semibold', 'Yu Gothic Light', '@Yu Gothic Light', 'Yu Gothic UI Light', '@Yu Gothic UI Light', 'Yu Gothic Medium', '@Yu Gothic Medium', 'Yu Gothic UI Semilight', '@Yu Gothic UI Semilight', '바탕', '@바탕', '바탕체', '@바탕체', '궁서', '@궁서', '궁서체', '@궁서체', '굴림', '@굴림', '굴림체', '@굴림체', '돋움', '@돋움', '돋움체', '@돋움체', 'HoloLens MDL2 Assets', '문체부 제목 바탕체', '@문체부 제목 바탕체', '문체부 제목 돋음체', '@문체부 제목 돋음체', '문체부 훈민정음체', '@문체부 훈민정음체', '문체부 궁체 정자체', '@문체부 궁체 정자체', '문체부 제목 돋음체', '@문체부 제목 돋음체', '문체부 궁체 흘림체', '@문체부 궁체 흘림체', '문체부 쓰기 정체', '@문체부 쓰기 정체', '문체부 쓰기 흘림체', '@문체부 쓰기 흘림체', '문체부 바탕체', '@문체부 바탕체', '함초롬바탕 확장B', '@함초롬바탕 확장B', 'NewJumja', 'HyhwpEQ', '@HyhwpEQ', '함초롬바탕', '@함초롬바탕', '함초롬바탕 확장', '@함초롬바탕 확장', '함초롬돋움', '@함초롬돋움', '함초롬돋움 확장', '@함초롬돋움 확장', 'Agency FB', 'Algerian', 'Book Antiqua', 'Arial Narrow', 'Arial Rounded MT Bold', 'Baskerville Old Face', 'Bauhaus 93', 'Bell MT', 'Bernard MT Condensed', 'Bodoni MT', 'Bodoni MT Black', 'Bodoni MT Condensed', 'Bodoni MT Poster Compressed', 'Bookman Old Style', 'Bradley Hand ITC', 'Britannic Bold', 'Berlin Sans FB', 'Berlin Sans FB Demi', 'Broadway', 'Brush Script MT', 'Bookshelf Symbol 7', 'Californian FB', 'Calisto MT', 'Castellar', 'Century Schoolbook', 'Centaur', 'Century', 'Chiller', 'Colonna MT', 'Cooper Black', 'Copperplate Gothic Bold', 'Copperplate Gothic Light', 'Curlz MT', 'Dubai', 'Dubai Light', 'Dubai Medium', 'Elephant', 'Engravers MT', 'Eras Bold ITC', 'Eras Demi ITC', 'Eras Light ITC', 'Eras Medium ITC', 'Felix Titling', 'Forte', 'Franklin Gothic Book', 'Franklin Gothic Demi', 'Franklin Gothic Demi Cond', 'Franklin Gothic Heavy', 'Franklin Gothic Medium Cond', 'Freestyle Script', 'French Script MT', 'Footlight MT Light', 'Garamond', 'Gigi', 'Gill Sans MT', 'Gill Sans MT Condensed', 'Gill Sans Ultra Bold Condensed', 'Gill Sans Ultra Bold', 'Gloucester MT Extra Condensed', 'Gill Sans MT Ext Condensed Bold', 'Century Gothic', 'Goudy Old Style', 'Goudy Stout', 'HY그래픽M', '@HY그래픽M', 'HY궁서B', '@HY궁서B', 'HY견고딕', '@HY견고딕', 'HY중고딕', '@HY중고딕', 'HY헤드라인M', '@HY헤드라인M', 'HY견명조', '@HY견명조', 'HY신명조', '@HY신명조', 'HY목각파임B', '@HY목각파임B', 'HY엽서L', '@HY엽서L', 'HY엽서M', '@HY엽서M', 'HY얕은샘물M', '@HY얕은샘물M', 'Harlow Solid Italic', 'Harrington', 'Haettenschweiler', '휴먼옛체', '@휴먼옛체', '휴먼편지체', '@휴먼편지체', '휴먼아미체', '@휴먼아미체', '휴먼매직체', '@휴먼매직체', '휴먼둥근헤드라인', '@휴먼둥근헤드라인', 'High Tower Text', 'Imprint MT Shadow', 'Informal Roman', 'Blackadder ITC', 'Edwardian Script ITC', 'Kristen ITC', 'Jokerman', 'Juice ITC', 'Kunstler Script', 'Wide Latin', 'Lucida Bright', 'Lucida Calligraphy', 'Lucida Fax', 'Lucida Handwriting', 'Lucida Sans', 'Lucida Sans Typewriter', 'Magneto', 'Maiandra GD', 'Matura MT Script Capitals', 'Mistral', 'Modern No. 20', 'Monotype Corsiva', 'MT Extra', '새굴림', '@새굴림', 'Niagara Engraved', 'Niagara Solid', 'OCR A Extended', 'Old English Text MT', 'Onyx', 'MS Outlook', 'Palace Script MT', 'Papyrus', 'Parchment', 'Perpetua', 'Perpetua Titling MT', 'Playbill', 'Poor Richard', 'Pristina', 'Rage Italic', 'Ravie', 'MS Reference Sans Serif', 'MS Reference Specialty', 'Rockwell Condensed', 'Rockwell', 'Rockwell Extra Bold', 'Script MT Bold', 'Showcard Gothic', 'Snap ITC', 'Stencil', 'Tw Cen MT', 'Tw Cen MT Condensed', 'Tw Cen MT Condensed Extra Bold', 'Tempus Sans ITC', 'Viner Hand ITC', 'Vivaldi', 'Vladimir Script', 'Wingdings 2', 'Wingdings 3', '휴먼모음T', '@휴먼모음T', '휴먼엑스포', '@휴먼엑스포', '나눔바른고딕', '@나눔바른고딕', '나눔바른고딕 Light', '@나눔바른고딕 Light', '나눔바른고딕 UltraLight', '@나눔바른고딕 UltraLight', '나눔바른고딕 옛한글', '@나눔바른고딕 옛한글', '나눔바른펜', '@나눔바른펜', '나눔손글씨 붓', '@나눔손글씨 붓', '나눔고딕', '@나눔고딕', '나눔고딕코딩', '@나눔고딕코딩', '나눔고딕 에코', '@나눔고딕 에코', '나눔고딕 ExtraBold', '@나눔고딕 ExtraBold', '나눔고딕 Light', '@나눔고딕 Light', '나눔명조', '@나눔명조', '나눔명조 에코', '@나눔명조 에코', '나눔명조 ExtraBold', '@나눔명조 ExtraBold', '나눔명조 옛한글', '@나눔명조 옛한글', '나눔손글씨 펜', '@나눔손글씨 펜', '나눔스퀘어_ac Bold', '@나눔스퀘어_ac Bold', '나눔스퀘어_ac ExtraBold', '@나눔스퀘어_ac ExtraBold', '나눔스퀘어_ac Light', '@나눔스퀘어_ac Light', '나눔스퀘어_ac', '@나눔스퀘어_ac', '나눔스퀘어 Bold', '@나눔스퀘어 Bold', '나눔스퀘어 ExtraBold', '@나눔스퀘어 ExtraBold', '나눔스퀘어 Light', '@나눔스퀘어 Light', '나눔스퀘어라운드 Bold', '@나눔스퀘어라운드 Bold', '나눔스퀘어라운드 ExtraBold', '@나눔스퀘어라운드 ExtraBold', '나눔스퀘어라운드 Light', '@나눔스퀘어라운드 Light', '나눔스퀘어라운드 Regular', '@나눔스퀘어라운드 Regular', 'DejaVu Sans Mono', '은 돋움', '@은 돋움', '은 바탕', '@은 바탕', 'KoPub돋움체 Medium', '@KoPub돋움체 Medium', 'KoPub돋움체 Bold', '@KoPub돋움체 Bold', 'KoPub바탕체 Light', '@KoPub바탕체 Light', 'KoPub바탕체 Bold', '@KoPub바탕체 Bold', 'a뉴공서M', '@a뉴공서M', 'a누리동실', '@a누리동실', 'a유리구슬', '@a유리구슬', 'a카리스마', '@a카리스마', '양재난초체M', '@양재난초체M', 'nukamiso2004', '@nukamiso2004', 'cinecaption', '@cinecaption', '62570체', '@62570체', 'a고운글씨M', '@a고운글씨M', 'a정훈시대M', '@a정훈시대M', '교보 손글씨 2019', '@교보 손글씨 2019', '교보 손글씨 2020 박도연', '@교보 손글씨 2020 박도연', '교보 손글씨 2021 성지영', '@교보 손글씨 2021 성지영', 'Cascadia Code ExtraLight', 'Cascadia Code Light', 'Cascadia Code SemiLight', 'Cascadia Code', 'Cascadia Code SemiBold', 'Cascadia Mono ExtraLight', 'Cascadia Mono Light', 'Cascadia Mono SemiLight', 'Cascadia Mono', 'Cascadia Mono SemiBold')
(newvenv) PS C:\Workspaces\Dev\Python\newvenv>
```

라벨 표시 위치

- 라벨은 왼쪽 위 모서리를 원점(0, 0)으로 하여 계산함

버튼 배치하기

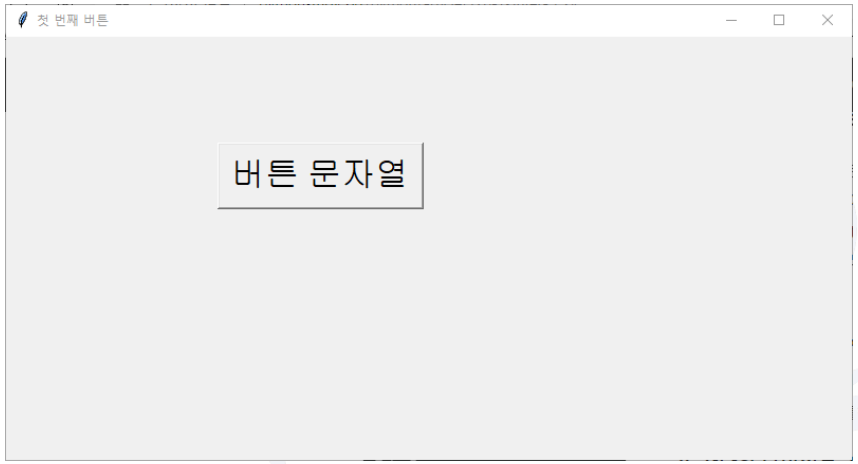
- 버튼: Button()
- 배치: place()

버튼 변수명=tkinter.Button(윈도우 객체, text="버튼 문자열", font=("폰트 명", 폰트 크기))

버튼 변수명.place(x=X 좌표, y=Y 좌표)

gui005.py

```
newvenv > gui005.py > ...
1   import tkinter
2
3   root = tkinter.Tk()
4   root.title("첫 번째 버튼")
5   root.geometry("800x400")
6
7   button = tkinter.Button(root, text="버튼 문자열", font=("Times New Roman", 24))
8   button.place(x=200, y=100)
9
10  root.mainloop()
11
```



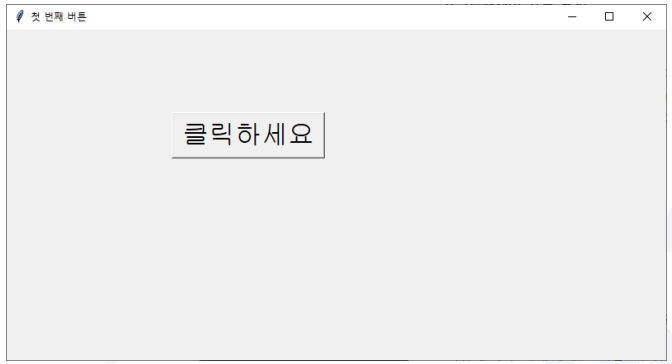
버튼 클릭 시 반응

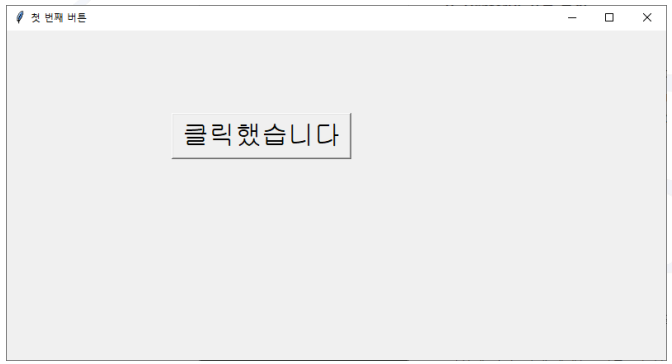
- 버튼을 클릭했을 때의 처리를 함수로 정의
- 버튼 생성 식 안에 “command=함수”를 입력
- 버튼이 클릭되면 해당 함수 실행

gui005.py | gui006.py

newvenv > gui006.py > ...

```python
import tkinter

def click_btn():
    button["text"] = "클릭했습니다"

root = tkinter.Tk()
root.title("첫 번째 버튼")
root.geometry("800x400")

button = tkinter.Button(root, text="클릭하세요", font=("System", 24), command=click_btn)
button.place(x=200, y=100)

root.mainloop()
```





- 캔버스 사용하기

캔버스 배치

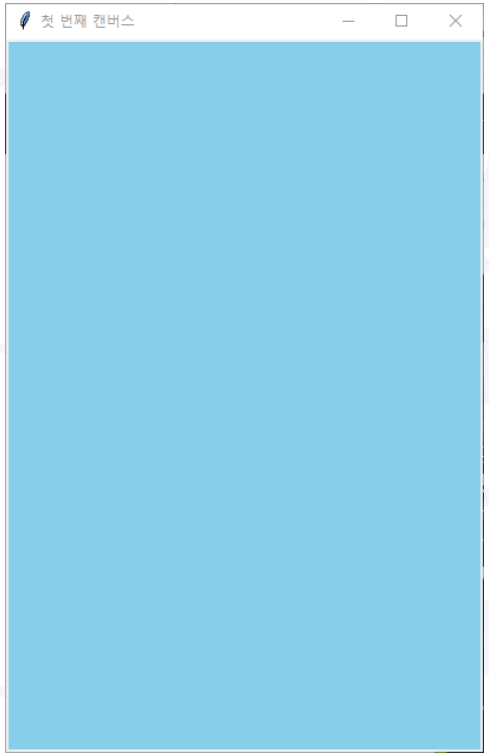
- 캔버스
  - 이미지나 도형을 그리는 GUI 요소
- 캔버스 생성: Canvas()
- 배치: pack(), place()

캔버스 변수명=tkinter.Canvas(윈도우 객체, width=폭, height=높이, bg="배경색")

- 배경색은 red, green, blue 등의 영어 단어나 16진수 값으로 지정 가능
- bg="배경색"은 생략 가능

gui007.py

```
newvenv > gui007.py > ...
1  import tkinter
2
3  root = tkinter.Tk()
4  root.title("첫 번째 캔버스")
5
6  canvas = tkinter.Canvas(root, width=400, height=600, bg="skyblue")
7  canvas.pack()
8  root.mainloop()
9
```



캔버스에 이미지 표시하기

- 이미지 로드: PhotoImage()
- 이미지 그리기: create_image()
  - create_image()에서 지정하는 좌표는 **캠퍼스의 정 중앙**이 원점임

```python
import tkinter

root = tkinter.Tk()
root.title("첫 번째 캔버스")

canvas = tkinter.Canvas(root, width=400, height=800)
canvas.pack()
my_image = tkinter.PhotoImage(file="./images/girl_001.png")
canvas.create_image(200, 400, image=my_image)
root.mainloop()
```

캔버스의 크기가 400x800 이므로 원점은 (200, 400)

이미지 출처: Pixabay

도형 그리기

- **Tkinter에서 제공하는 도형 그리기 명령**
  - **직선: create_line(x1, y1, x2, y2, fill="색" width="선 굵기")**
    - 곡선 그리기: 3점 이상 지정 시, smooth=True 로 지정하면 곡선을 그림
  - **사각형: create_rectangle (x1, y1, x2, y2, fill="내부 색" outline="테두리 색" width="테두리 굵기")**
  - **타원형: create_oval(x1, y1, x2, y2, fill="내부 색", outline="테두리 색", width="테두리 굵기")**

도형 그리기

- **다각형: create_polygon(x1, y1, x2, y2, x3, y3, ..., fill="내부 색", outline="테두리 색", width="테두리 굵기")**

- **원호: create_arc(x1, y1, x2, y2, fill="내부 색", outline="테두리 색", start="시작 각도", extent="그릴 횟수", style=tkinter.***)**
  - ***에는 PIESLCIE, CHORD, ARC를 지정할 수 있음

도형 그리기

gui009.py

```
newvenv > gui009.py > ...
```

```python
import tkinter

root = tkinter.Tk()
root.title("캔버스에 도형 그리기")
root.geometry("500x400")

canvas = tkinter.Canvas(root, width=500, height=400, bg="white")
canvas.pack()

# 텍스트 그리기
canvas.create_text(250, 25, text="문자열", fill="green", font=("System", 24))

# 직선 그리기
canvas.create_line(30, 30, 70, 80, fill="navy", width=5)

# 곡선 그리기
canvas.create_line(120, 20, 80, 50, 200, 80, 140, 120, fill="blue", smooth=True)

# 사각형 그리기
canvas.create_rectangle(40, 140, 160, 200, fill="lime")
canvas.create_rectangle(60, 240, 120, 360, fill="pink", outline="red", width=5)

# 타원 그리기
canvas.create_oval(250-40, 100-40, 250+40, 100+40, fill="silver", outline="purple")
canvas.create_oval(250-80, 200-40, 250+80, 200+40, fill="cyan", width=0)

# 다각형 그리기
canvas.create_polygon(250, 250, 150, 350, 350, 350, fill="magenta", width=0)

# 원호 그리기
canvas.create_arc(400-50, 100-50, 400+50, 100+50, fill="yellow", start=30, extent=300)
canvas.create_arc(400-50, 250-50, 400+50, 250+50, fill="gold", start=0, extent=120, style=tkinter.CHORD)
canvas.create_arc(400-50, 350-50, 400+50, 350+50, fill="orange", start=0, extent=120, style=tkinter.ARC)

root.mainloop()
```

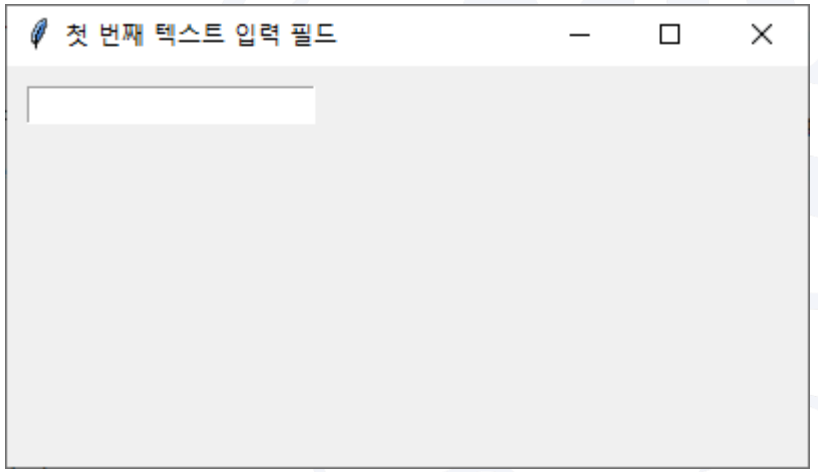
- 다양한 위젯(Widget) 배치하고 사용하기

텍스트 입력 필드

- 텍스트 입력 필드
  - 1행 텍스트 입력 필드: Entry()
  - 여러 행 텍스트 입력 필드: Text()

gui010.py

```
newvenv > gui010.py > ...
1   import tkinter
2
3   root = tkinter.Tk()
4   root.title("첫 번째 텍스트 입력 필드")
5   root.geometry("400x200")
6
7   entry = tkinter.Entry(width=20)
8   entry.place(x=10, y=10)
9
10  root.mainloop()
11
```



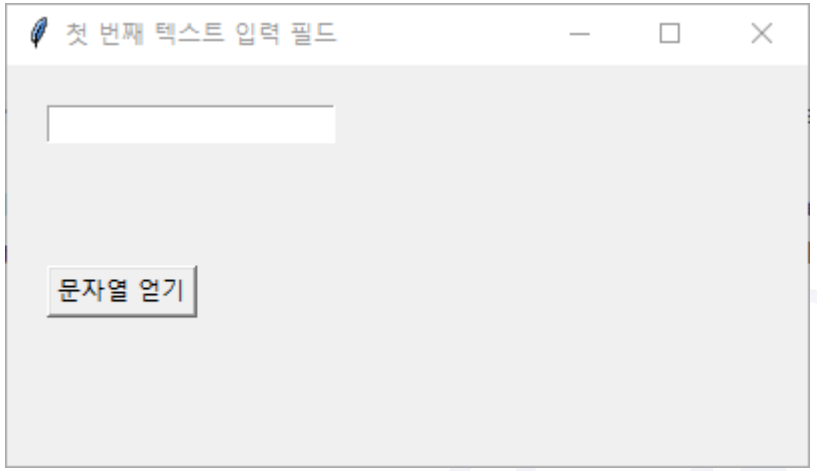
텍스트 입력 필드

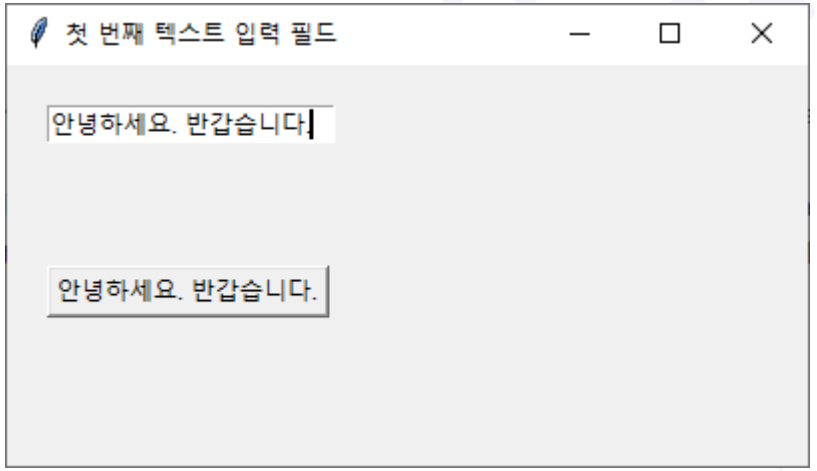
- **Entry 내 문자열 조작하기**
  - **Entry 내 문자열 얻기: get()**

gui010.py | gui011.py

newvenv > gui011.py > ...

```
1  import tkinter
2
3  def click_btn():
4      txt = entry.get()
5      button["text"] = txt
6
7  root = tkinter.Tk()
8  root.title("첫 번째 텍스트 입력 필드")
9  root.geometry("400x200")
10
11 entry = tkinter.Entry(width=20)
12 entry.place(x=20, y=20)
13
14 button = tkinter.Button(text="문자열 얻기", command=click_btn)
15 button.place(x=20, y=100)
16
17 root.mainloop()
18
```





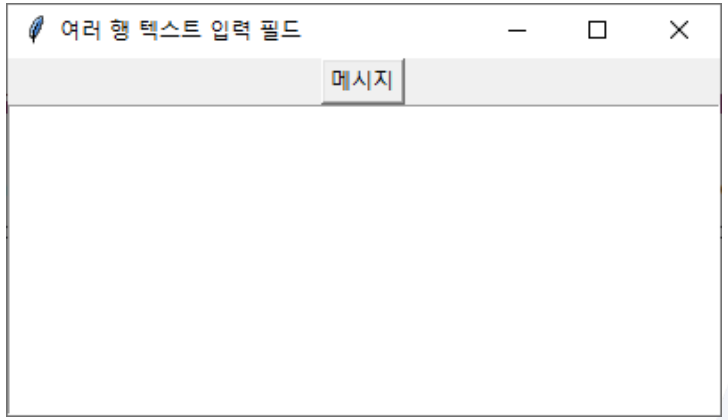
- Entry 내의 문자열을 삭제할 때는 delete()
- Entry 내에 문자열을 삽입할 때는 insert()

텍스트 입력 필드

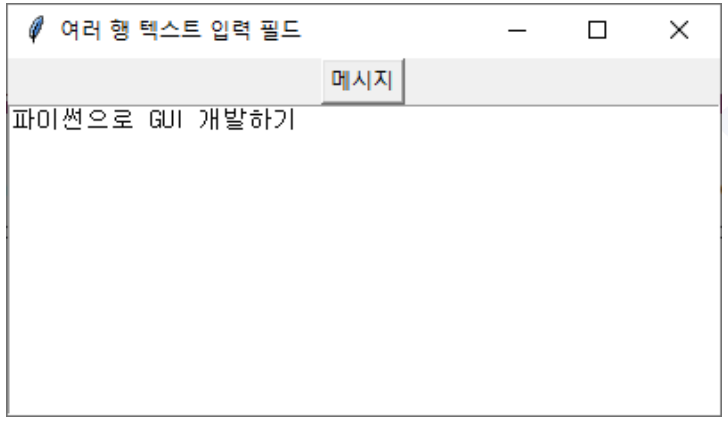
- 여러 행 텍스트 입력 필드

```python
import tkinter

def click_btn():
    text.insert(tkinter.END, "파이썬으로 GUI 개발하기")

root = tkinter.Tk()
root.title("여러 행 텍스트 입력 필드")
root.geometry("400x200")

button = tkinter.Button(text="메시지", command=click_btn)
button.pack()

text = tkinter.Text()
text.pack()

root.mainloop()
```

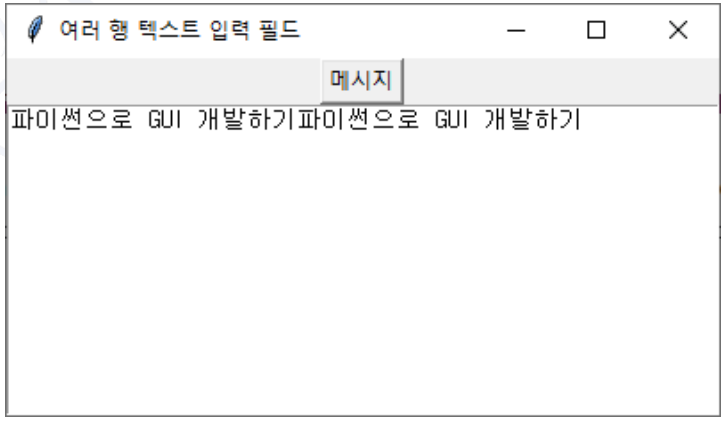
text.insert()에서
tkinter.END는 새로 입력하는 문장을
기존 문장의 끝에 추가하라는 의미



pack() 대신
place()함수를 사용해도 됨



One Click !!!



Two Click !!!

텍스트 입력 필드

- **Text에 입력된 문자열을 얻으려면**
  - **get(시작 위치, 종료 위치) 함수 사용**
    - get("1.0", "end-1c") → 입력필드의 전체 문자열을 얻을 때
      - 1.0: 1행 0번째 문자(즉, 첫번째 문자)
      - end-1c: end만 지정하면 가장 마지막 위치의 다음 위치가 되므로 거기에서 1문자 앞

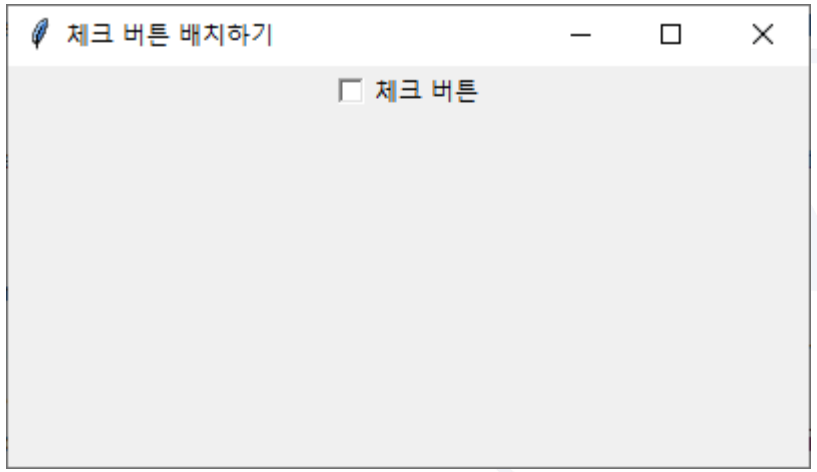
- **Text에 입력된 문자열을 삭제하려면**
  - **delete(시작 위치, 종료 위치) 함수 사용**

체크 버튼

- 체크 버튼 배치하기
  - 체크 버튼 생성: Checkbutton()

gui013.py

```
newvenv > gui013.py > ...
1   import tkinter
2
3   root = tkinter.Tk()
4   root.title("체크 버튼 배치하기")
5   root.geometry("400x200")
6
7   cbtn = tkinter.Checkbutton(text="체크 버튼")
8   cbtn.pack()
9
10  root.mainloop()
11
```



체크 버튼

- 체크 값 설정하기
  - 체크 값 → True/False → BooleanVar()

gui013.py gui014.py

newvenv > gui014.py > ...

```
1   import tkinter
2
3   root = tkinter.Tk()
4   root.title("처음부터 체크된 상태 만들기")
5   root.geometry("400x200")
6
7   cvalue = tkinter.BooleanVar()
8   cvalue.set(True)
9
10  cbtn = tkinter.Checkbutton(text="체크 버튼", variable=cvalue)
11  cbtn.pack()
12
13  root.mainloop()
14
```

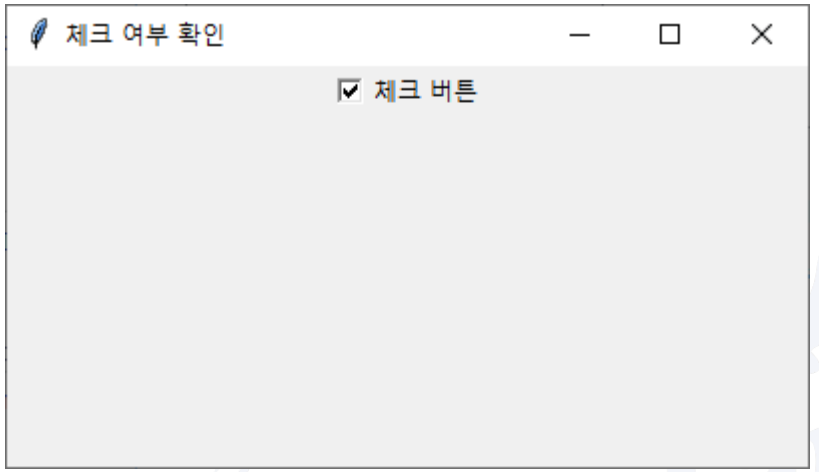
체크 버튼

- 체크 여부 확인하기
  - 체크 값 → True/False → BooleanVar()
  - BooleanVar 객체에서 get()으로 확인

gui013.py gui014.py gui015.py

newvenv > gui015.py > ...

```python
import tkinter

def check():
    if cvalue.get() == True:
        print("체크되어 있습니다")
    else:
        print("체크되어 있지 않습니다")

root = tkinter.Tk()
root.title("체크 여부 확인")
root.geometry("400x200")

cvalue = tkinter.BooleanVar()
cvalue.set(False)

cbtn = tkinter.Checkbutton(text="체크 버튼", variable=cvalue, command=check)
cbtn.pack()

root.mainloop()
```



터미널 JUPYTER 디버그 콘솔 문제 출력

```
(newvenv) PS C:\Workspaces\Dev\Python\newvenv> python gui015.py
체크되어 있습니다
체크되어 있지 않습니다
체크되어 있습니다
체크되어 있지 않습니다
체크되어 있습니다
```

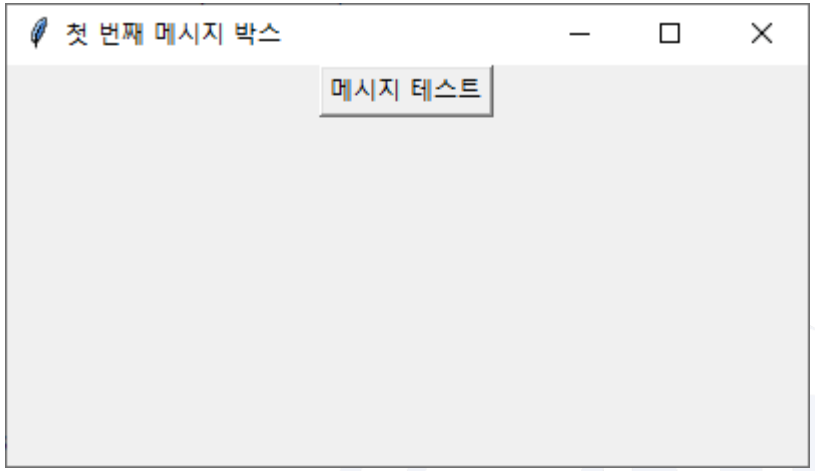
메시지 박스

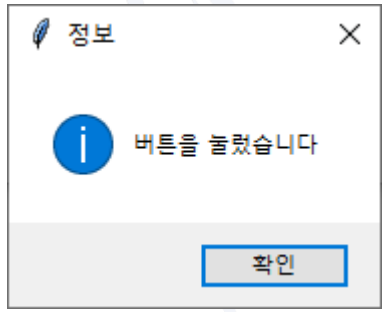
- **메시지 박스 표시하기**
  - **tkinter.messagebox 모듈 임포트**
  - **메시지 박스 표시 명령어**
    - showinfo(): 정보를 표시하는 메시지 박스
    - showwarning(): 경고를 표시하는 메시지 박스
    - showerror(): 에러를 표시하는 메시지 박스
    - askyesno(): '예', '아니오' 버튼이 있는 메시지 박스
    - askokcancle(): 'OK', '취소' 버튼이 있는 메시지 박스

메시지 박스

gui016.py

```
newvenv > gui016.py > ...
1  import tkinter
2  import tkinter.messagebox
3
4  def click_btn():
5      tkinter.messagebox.showinfo("정보", "버튼을 눌렀습니다")
6
7  root = tkinter.Tk()
8  root.title("첫 번째 메시지 박스")
9  root.geometry("400x200")
10
11 btn = tkinter.Button(text="메시지 테스트", command=click_btn)
12 btn.pack()
13
14 root.mainloop()
15
```





Tkinter의 다양한 위젯과 그 역할

- Tk
  - 여러 위젯과 레이아웃을 적용할 윈도우 창의 화면 생성
  - mainloop()를 실행해야 윈도우 창이 표시됨
  - mainloop(): 마우스, 화면에서 발생하는 이벤트 감지 역할
  - geometry(): 화면의 크기, 위치 조절
- Button
  - 확인 버튼과 같은 버튼 기능 구현
  - Command 속성을 이용하여 이벤트 처리 함수를 바로 적용할 수 있음

Tkinter의 다양한 위젯과 그 역할

- **Label**
  - 화면 상에 텍스트로 된 설명 출력 시 사용
  - 주로 입력 Entry 앞에 지문을 출력하는 용도로 사용됨
- **Entry**
  - 텍스트를 한 라인씩 입력 받을 수 있는 라인 텍스트 입력 필드
- **Text**
  - 여러 줄 텍스트를 입력받을 수 있는 텍스트 영역 생성

Tkinter의 다양한 위젯과 그 역할

- Canvas
  - 그래픽 화면 지원
  - 도형과 선을 이용하여 그래프를 그리거나 사용자 정의 위젯을 만들 때 사용
- PhotoImage
  - 그림이나 사진을 붙일 때 사용
  - PhotoImage 객체를 Label에 넣어서 화면에 붙일 수도 있음

Tkinter의 다양한 위젯과 그 역할

- Checkbutton: 여러 옵션 중 하나 또는 그 이상의 옵션을 선택하는 체크박스 위젯
- Radiobutton: 여러 옵션 중 하나만 선택하는 라디오 버튼 위젯
- Listbox: 리스트 박스 위젯
- Scale: 슬라이드 바 위젯
- Scrollbar: 스크롤 바 위젯
- Spinbox: 스핀 박스 위젯

Tkinter의 다양한 위젯과 그 역할

- ttk.Combobox: 드롭다운 목록 표시하는 콤보박스 위젯
- ttk.Notebook: 탭이 붙은 여러 페이지를 겹쳐서 적용할 수 있음
- ttk.Progressbar:
  - 다운로드나 실행 중인 프로그램의 진행 사항을 표시하는 바 생성
- ttk.Treeview: 행과 열로 구성된 표 생성

- Frame: 컨테이너나 다른 위젯을 그룹화할 때 사용
- LabelFrame: Label, Button, Entry 등의 위젯의 그룹 테두리 생성
- PanedWindow: 기존 레이아웃 안에 다른 레이아웃을 적용하거나 다른 위젯을 붙이기 위해 사용하는 위젯
- Menu: Tk 윈도우에 메뉴 바 생성
- Menubutton: 메뉴 바에 부탁될 메뉴 버튼 생성

Tkinter의 다양한 위젯과 그 역할

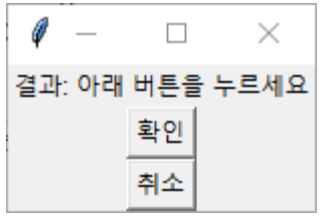
- Message
  - Label처럼 텍스트를 화면에 표시할 때 사용
  - 자동 래핑 기능이 있다는 점이 Label과 다름
  - MessageBox와 다름(메시지 박스는 위젯이 아님)
- Toplevel: Tk에서 새로운 윈도우를 추가로 만들 때 사용

Frame을 이용한 위젯 배치

gui017.py

newvenv > gui017.py > ...

```
1   from tkinter import Tk, Label, Button
2   import tkinter.messagebox
3
4   win = Tk()
5   win.title("Frame을 이용한 위젯 배치")
6
7   lbl = Label(win, text="결과: 아래 버튼을 누르세요")
8   btn1 = Button(win, text="확인")
9   btn2 = Button(win, text="취소")
10
11  lbl.pack()
12  btn1.pack()
13  btn2.pack()
14
15  if __name__ == '__main__':
16      win.mainloop()
17
```




```
if __name__ == '__main__':
    ... 코드 ...
    ... 코드 ...
```

해당 모듈이
임포트된 경우가 아니라
인터프리터에서 직접 실행된 경우에만
if문 이하의 코드를 돌리라는 명령

인터프리터에서 직접 실행하면 __name__ 변수에
"__main__"이 담겨서 전달됨
모듈에서 임포트되면 __name__ 변수에
"executeThisModule"이 담겨서 전달됨

__name__ : interpreter가 실행 전에 만들어 둔
글로벌 변수

다른 모듈에서 임포트된 경우에는 내부에 구현된
기능(함수)만 뽑아서 사용하겠다는 의미이므로
전체 루프를 실행할 필요가 없음

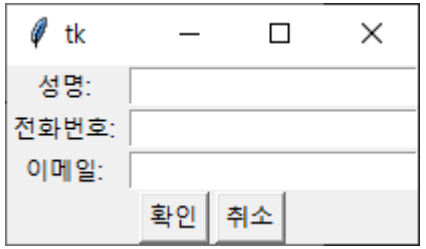
배치 관리자를 이용한 위젯 배치

- **배치 관리자의 종류**
  - **pack: 압축 배치 관리자**
    - 한 줄로 이어붙이는 방식으로 배치. 불필요한 공간을 없앨 수 있음
  - **grid: 격자 배치 관리자**
    - 행, 열의 위치를 지정해서 배치 (row, column 속성을 이용해서 지정된 위치에 배치)
    - rowspan, columnspan 속성을 이용하여 셀 병함도 가능함
  - **place: 절대 배치 관리자**
    - 절대 좌표를 이용해서 강제로 위치 지정 (pack, grid는 상대적 방식)
    - 단점: 윈도우의 크기에 따라 위젯이 변경되지 않음

배치 관리자를 이용한 위젯 배치

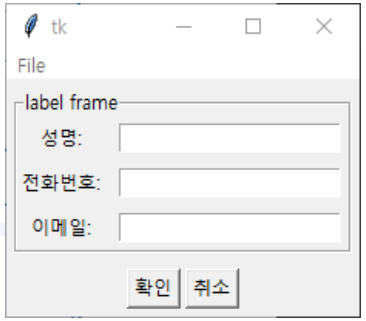
gui018.py

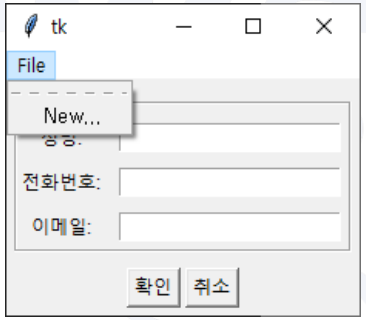
```
newvenv > gui018.py > ...
1  from tkinter import Tk, PanedWindow
2  from tkinter import Label, Entry, Button
3
4  win = Tk()
5
6  lbl_name = Label(win, text="성명: ")
7  lbl_phone = Label(win, text="전화번호: ")
8  lbl_email = Label(win, text="이메일: ")
9
10 entry_name = Entry(win)
11 entry_phone = Entry(win)
12 entry_email = Entry(win)
13
14 lbl_name.grid(row=0, column=0)
15 entry_name.grid(row=0, column=1)
16 lbl_phone.grid(row=1, column=0)
17 entry_phone.grid(row=1, column=1)
18 lbl_email.grid(row=2, column=0)
19 entry_email.grid(row=2, column=1)
20
21 panedwindow = PanedWindow(relief="raised", bd=0)
22 panedwindow.grid(row=3, column=0, columnspan=2)
23
24 btn_ok = Button(panedwindow, text="확인")
25 btn_cancle = Button(panedwindow, text="취소")
26
27 panedwindow.add(btn_ok)
28 panedwindow.add(btn_cancle)
29
30 if __name__ == '__main__':
31     win.mainloop()
32
```



gui018.py gui019.py

```
newvenv > gui019.py > ...
1  from tkinter import Tk, PanedWindow, Menu, LabelFrame
2  from tkinter import Label, Entry, Button
3
4  win = Tk()
5
6  menu_area = Menu(win)
7  win.configure(menu=menu_area)
8
9  menu1 = Menu(menu_area)
10 menu1.add_command(label="New...")
11 menu_area.add_cascade(label="File", menu=menu1)
12
13 label_frame = LabelFrame(win, text="label frame")
14 label_frame.grid(row=0, column=0, padx=5, pady=5)
15
16 lbl_name = Label(label_frame, text="성명: ")
17 lbl_phone = Label(label_frame, text="전화번호: ")
18 lbl_email = Label(label_frame, text="이메일: ")
19
20 entry_name = Entry(label_frame)
21 entry_phone = Entry(label_frame)
22 entry_email = Entry(label_frame)
23
24 lbl_name.grid(row=0, column=0)
25 entry_name.grid(row=0, column=1, padx=5, pady=5)
26 lbl_phone.grid(row=1, column=0)
27 entry_phone.grid(row=1, column=1, padx=5, pady=5)
28 lbl_email.grid(row=2, column=0)
29 entry_email.grid(row=2, column=1, padx=5, pady=5)
30
31 panedwindow = PanedWindow(relief="raised", bd=0)
32 panedwindow.grid(row=3, column=0, columnspan=2, padx=5, pady=5)
33
34 btn_ok = Button(panedwindow, text="확인")
35 btn_cancle = Button(panedwindow, text="취소")
36
37 panedwindow.add(btn_ok)
38 panedwindow.add(btn_cancle)
39
40 if __name__ == '__main__':
41     win.mainloop()
42
```





- 미니 프로젝트

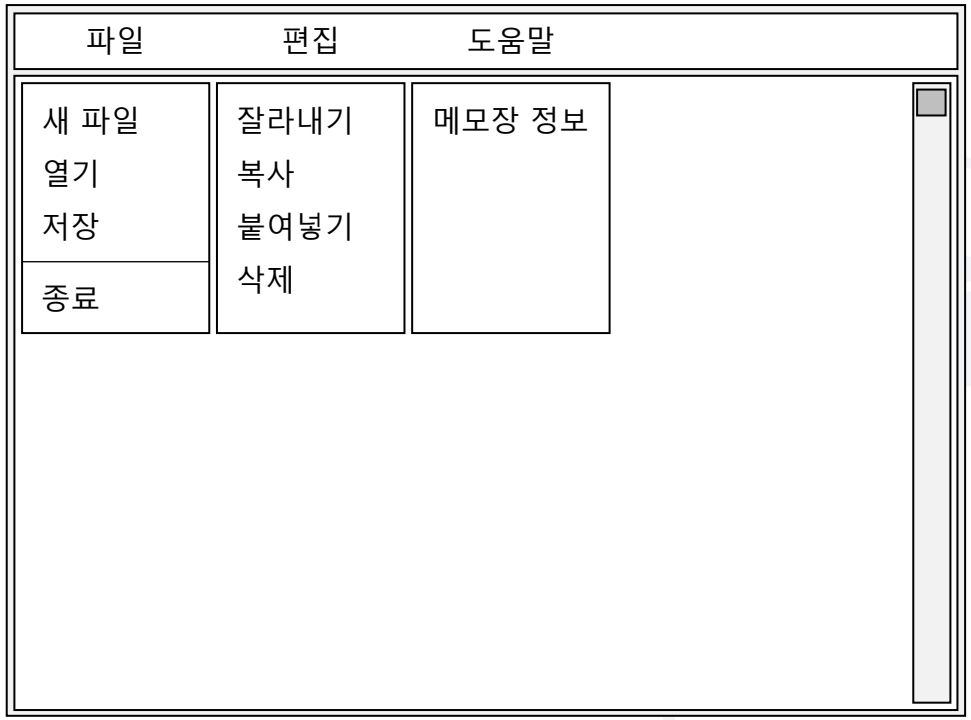
미니 프로젝트

- 메모장 개발
  - GUI 구성
  - 메모장 기능 구현

- 계산기 개발
  - GUI 구성
  - 계산기 기능 구현

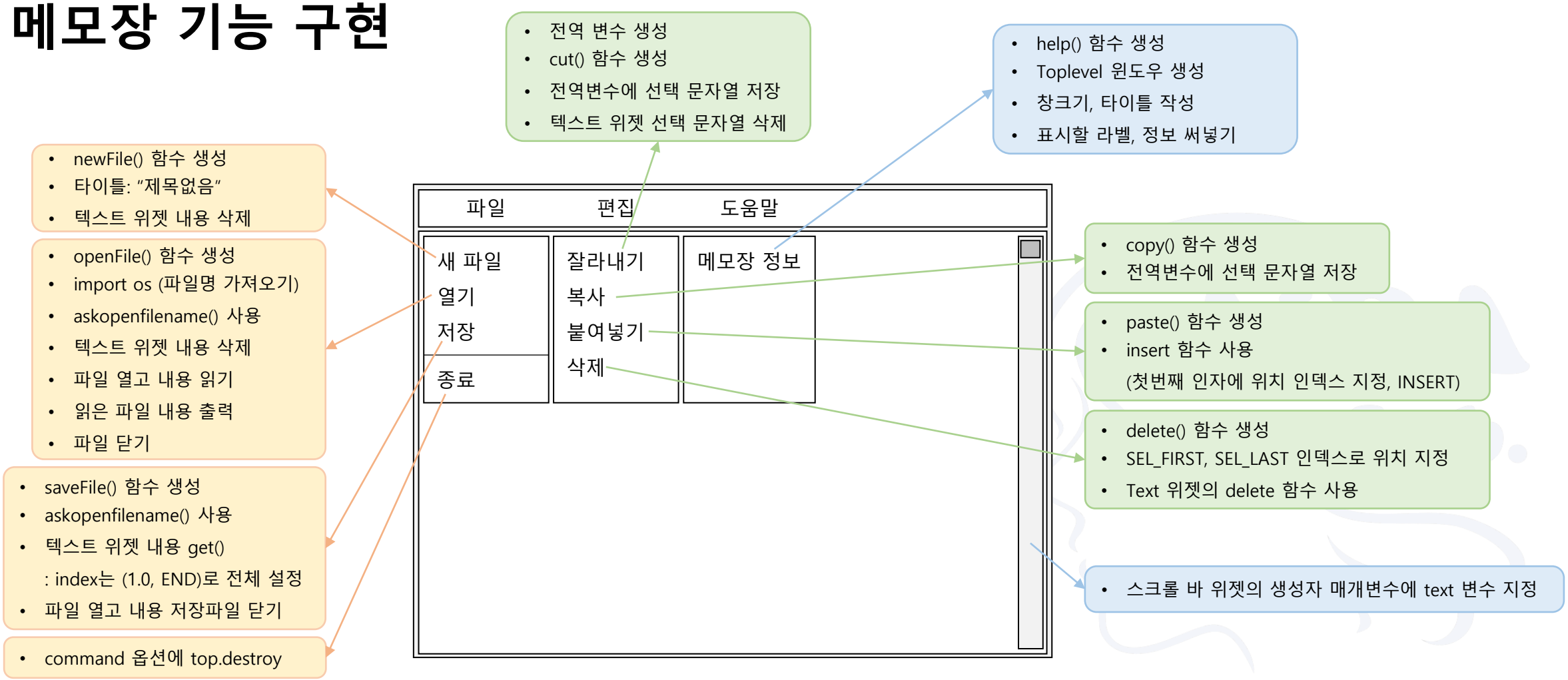
메모장 개발

- GUI 구성하기
  - 컴포넌트 배치
    - 메뉴
    - 텍스트 영역
    - 스크롤 바
  - 메뉴 구성
    - 메뉴 바 그리기
    - 하위메뉴 그리기



메모장 개발

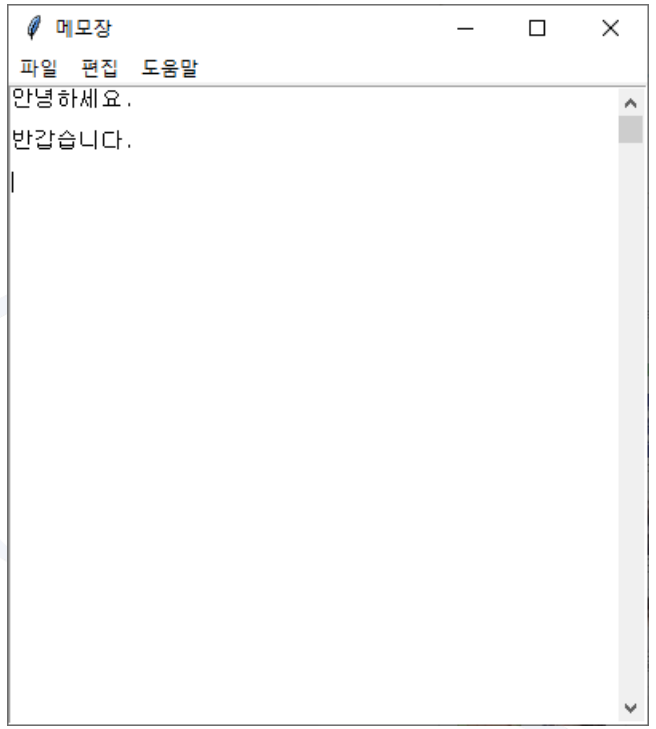
- 메모장 기능 구현



메모장 개발

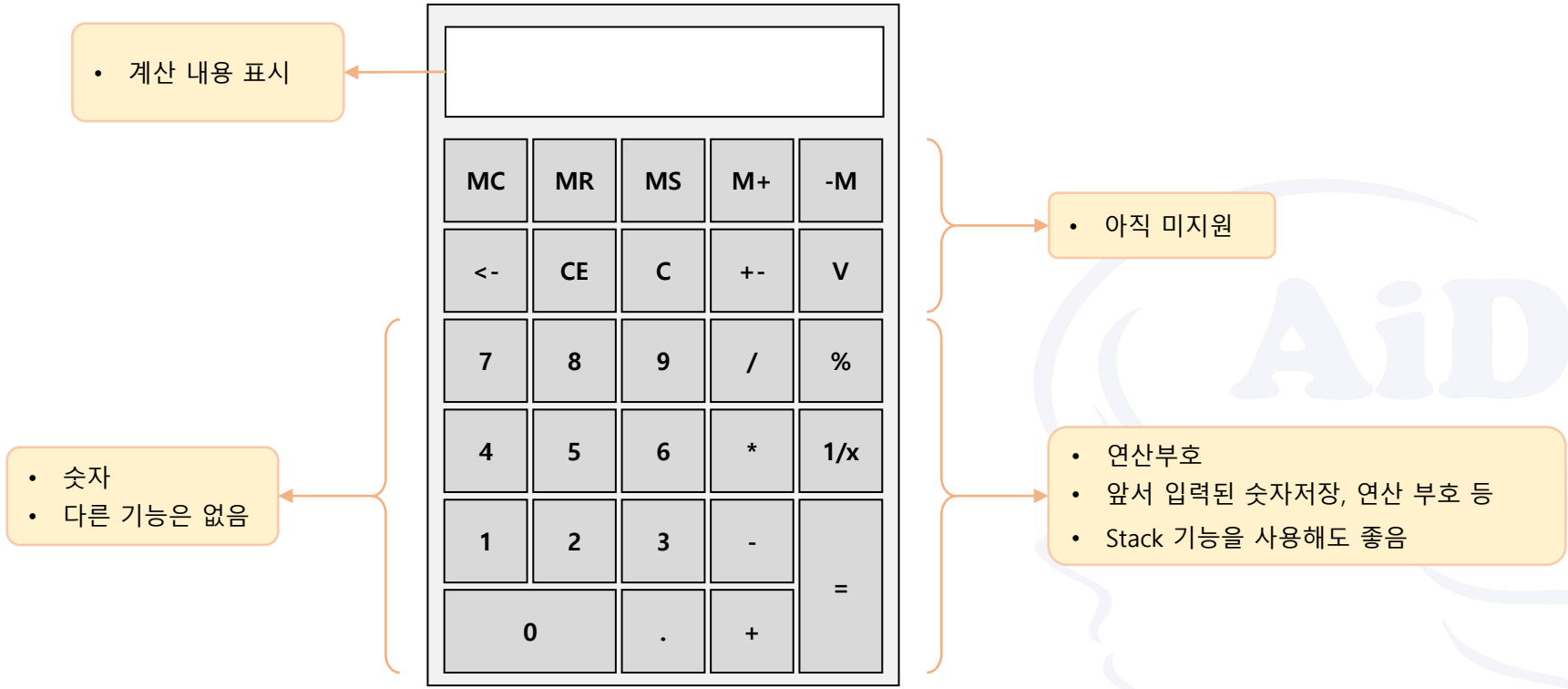
memopad03.py

```
newvenv > memopad03.py > ...
from tkinter import *
import os
from tkinter.filedialog import *

es = ""

def newFile():
    top.title("제목없음- 메모장")
    file = None
    ta.delete(1.0,END)

def openFile():
    file = askopenfilename(title = "파일 선택", filetypes = (("텍스트 파일", "*.txt"),("모든 파일", "*.*")))
    top.title(os.path.basename(file) + " - 메모장")
    ta.delete(1.0, END)
    f = open(file,"r")
    ta.insert(1.0,f.read())
    f.close()

def saveFile():
    f = asksaveasfile(mode = "w", defaultextension=".txt")
    if f is None:
        return
    ts = str(ta.get(1.0, END))
    f.write(ts)
    f.close()

def cut():
    global es
    es = ta.get(SEL_FIRST, SEL_LAST)
    ta.delete(SEL_FIRST, SEL_LAST)

def copy():
    global es
    es = ta.get(SEL_FIRST, SEL_LAST)

def paste():
    global es
    ta.insert(INSERT, es)

def delete():
    ta.delete(SEL_FIRST, SEL_LAST)

def help():
    he = Toplevel(top)
    he.geometry("200x200")
    he.title("정보")
    lb = Label(he, text = "메모장 버전 1.0\n 파이썬으로 만든 메모장입니다^^")
    lb.pack()

top = Tk()
top.title("메모장")
top.geometry("400x400")


ta = Text(top)
sb = Scrollbar(ta)
sb.config(command = ta.yview)
top.grid_rowconfigure(0, weight=1)
top.grid_columnconfigure(0, weight=1)
sb.pack(side = RIGHT, fill = Y)
ta.grid(sticky = N + E + S + W)

file = None


mb = Menu(top)
fi = Menu(mb, tearoff=0)
fi.add_command(label="새파일", command=newFile)
fi.add_command(label="열기", command=openFile)
fi.add_command(label="저장", command=saveFile)
fi.add_separator()
fi.add_command(label="종료", command=top.destroy)
mb.add_cascade(label="파일", menu=fi)

e = Menu(mb, tearoff=0)
e.add_command(label="잘라내기", command=cut)
e.add_command(label="복사", command=copy)
e.add_command(label="붙이기", command=paste)
e.add_command(label="삭제", command=delete)
mb.add_cascade(label="편집", menu=e)

h = Menu(mb, tearoff=0)
h.add_command(label="메모장 정보", command = help)
mb.add_cascade(label="도움말", menu=h)

top.config(menu=mb)

top.mainloop()
```



계산기 개발

- GUI 구성하기



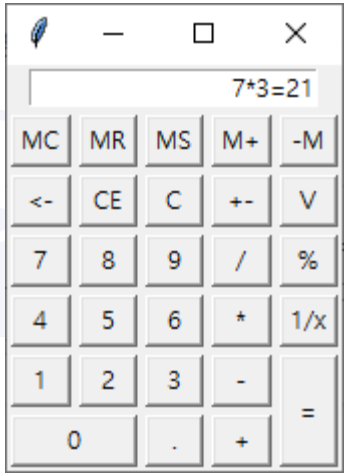
계산기 개발

calc03.py

```
newvenv > calc03.py > calc
```

```python
from tkinter import Tk, Button, Entry, END

win = Tk()

opers = []
nums = []
numStr = ''

def calc(target):
    ch = target['text']
    global opers, nums, numStr

    if len(ch) == 1:
        if ch != 'C' and ch != '%' and ch != 'v':
            txt.insert(END, ch)

        if ord(ch) >= 48 and ord(ch) <= 57:
            numStr += ch

        if ch == '+' or ch == '-' or ch == '*' or ch == '/':
            nums.append(numStr)
            opers.append(ch)
            numStr = ''

        if ch == '=':
            nums.append(numStr)
            result = int(nums[0])
            for i, oper in enumerate(opers):
                if oper == '+':
                    result += int(nums[i+1])
                elif oper == '-':
                    result -= int(nums[i+1])
                elif oper == '*':
                    result *= int(nums[i+1])
                elif oper == '/':
                    result //= int(nums[i+1])

            txt.insert(END, str(result))

    return 0

txt = Entry(win, justify="right")
txt.grid(row=0, column=0, columnspan=5, pady=2)
```

```python
btns = [
    [Button(win, text="MC"), Button(win, text="MR"), Button(win, text="MS"), Button(win, text="M+"), Button(win, text="-M")],
    [Button(win, text="<-"), Button(win, text="CE"), Button(win, text="C"),  Button(win, text="+-"), Button(win, text="V")],
    [Button(win, text="7"),  Button(win, text="8"),  Button(win, text="9"),  Button(win, text="/"),  Button(win, text="%")],
    [Button(win, text="4"),  Button(win, text="5"),  Button(win, text="6"),  Button(win, text="*"),  Button(win, text="1/x")],
    [Button(win, text="1"),  Button(win, text="2"),  Button(win, text="3"),  Button(win, text="-"),  Button(win, text="=")],
    [Button(win, text="0"),  Button(win, text="."),  Button(win, text="+")]
]

for btnArr in btns:
    for btn in btnArr:
        def clickEvent(target=btn):
            calc(target)
        btn.config(command=clickEvent)

for i in range(1, 6):
    for j in range(5):
        if i==6 and j==3:
            break
        if i==5 and j==4:
            btns[i-1][j].grid(row=i, column=j, rowspan=2, padx=2, pady=2, sticky='wens')
        else:
            btns[i-1][j].grid(row=i, column=j, padx=2, pady=2, sticky='wens')

btns[5][0].grid(row=6, column=0, columnspan=2, padx=2, pady=2, sticky='wens')
btns[5][1].grid(row=6, column=2, padx=2, pady=2, sticky='wens')
btns[5][2].grid(row=6, column=3, padx=2, pady=2, sticky='wens')

if __name__ == '__main__':
    win.mainloop()
```



참고 자료

- 파이썬으로 배우는 게임 개발:입문편 (히로세 츠요시 지음 / 김연수 옮김 | 제이펍)
- 400제로 배우는 파이썬 입문 (김범준 지음 | 심통)