Day - 34

**Type-Hint**

**&**

**Rebuild Question-game with GUI**

API-application, type-hinting, Creating a GUI Quiz App

**34.1 Implementing API**

* Separate parameters from this URL:

*https://opentdb.com/api.php?amount=10&type=boolean*

**from** question\_model **import** Question

#*from data import question\_data*

**from** quiz\_brain **import** QuizBrain

**import** requests

#*get questions from API*

#*Separate parameters from this URL:*

#*https://opentdb.com/api.php?amount=10&type=boolean*

#*"https://opentdb.com/api.php" is url and "amount=10&type=boolean" are parameters*

parMs = {

    "amount" : 10,

    "type" : "boolean"

}

api\_ques = requests**.get**(url = "https://opentdb.com/api.php", params = parMs)

question\_data = api\_ques**.json**()["results"]

* Html ENTITIES conversion:

**from** question\_model **import** Question

#*from data import question\_data*

**from** quiz\_brain **import** QuizBrain

**import** html

**import** requests

#*get questions from API*

#*Separate parameters from this URL:*

#*https://opentdb.com/api.php?amount=10&type=boolean*

#*"https://opentdb.com/api.php" is url and "amount=10&type=boolean" are parameters*

parMs = {

    "amount" : 10,

    "type" : "boolean"

}

api\_ques = requests**.get**(url = "https://opentdb.com/api.php", params = parMs)

api\_ques**.raise\_for\_status**()

question\_data = api\_ques**.json**()["results"]

#*Html code conversion: use "html" module and use unescape*

#*question\_text = html.unescape(question["question"])*

question\_bank = []

**for** question **in** question\_data:

    question\_text = html**.unescape**(question["question"])

    question\_answer = question["correct\_answer"]

    new\_question = **Question**(question\_text, question\_answer)

    question\_bank**.append**(new\_question)

quiz = **QuizBrain**(question\_bank)

**while** quiz**.still\_has\_questions**():

    quiz**.next\_question**()

**print**("You've completed the quiz")

**print**(f"Your final score was: {quiz**.**score}/{quiz**.**question\_number}")

#*python quiz\_main\_instructor.py*

* Using "self": No need to use "self" if we are in the same focus. But need to use it, if function is called in other function. For example notice the image files.

|  |  |
| --- | --- |
| " **self** " not used with image files | " **self** " is used with image files |
| **import** tkinter  THEME\_COLOR = "#375362"  **class** quiz\_ui:  **def** **\_\_init\_\_**(self):  **self.**winDow = tkinter**.Tk**()  **self.**winDow**.title**("Quizler")  **self.**winDow**.config**(padx =20, pady = 20, bg= THEME\_COLOR)    **self.**score\_label = tkinter**.Label**(text = "Score 0", fg = "white", bg = THEME\_COLOR)  **self.**score\_label**.grid**(column = 1, row = 0)  **self.**cAnvs = tkinter**.Canvas**(width = 300, height = 250)  **self.**cAnvs**.grid**(column = 0, row = 1, columnspan = 2, pady = 50)  **self.**question\_text = **self.**cAnvs**.create\_text**(                      150,                      125,                      text = "Some text will go here",                      fill = THEME\_COLOR,                      font = ("Arial", 24, "italic"))          #*always remember to use keywords: "file"*          rt\_btn\_img = tkinter**.PhotoImage**(file = "./images/true.png")          wrng\_btn\_img = tkinter**.PhotoImage**(file = "./images/false.png")  **self.**crrCT = tkinter**.Button**(image = rt\_btn\_img, bd = 0, highlightthickness=0)  **self.**crrCT**.config**(pady = 50)  **self.**crrCT**.grid**(column = 0, row = 2)  **self.**flSe = tkinter**.Button**(image = wrng\_btn\_img, bd = 0, highlightthickness=0)  **self.**flSe**.config**(pady = 50)  **self.**flSe**.grid**(column = 1, row = 2)  **self.**winDow**.mainloop**() | **import** tkinter  THEME\_COLOR = "#375362"  **class** quiz\_ui:  **def** **\_\_init\_\_**(self):  **self.**winDow = tkinter**.Tk**()  **self.**winDow**.title**("Quizler")  **self.**winDow**.config**(padx =20, pady = 20, bg= THEME\_COLOR)    **self.**score\_label = tkinter**.Label**(text = "Score 0", fg = "white", bg = THEME\_COLOR)  **self.**score\_label**.grid**(column = 1, row = 0)  **self.canV**()  **self.**winDow**.mainloop**()  **def** **canV**(self):  **self.**cAnvs = tkinter**.Canvas**(width = 300, height = 250)  **self.**cAnvs**.grid**(column = 0, row = 1, columnspan = 2, pady = 50)  **self.**question\_text = **self.**cAnvs**.create\_text**(                      150,                      125,                      text = "Some text will go here",                      fill = THEME\_COLOR,                      font = ("Arial", 24, "italic"))          #*always remember to use keywords: "file"*  **self.**rt\_btn\_img = tkinter**.PhotoImage**(file = "./images/true.png")  **self.**wrng\_btn\_img = tkinter**.PhotoImage**(file = "./images/false.png")  **self.**crrCT = tkinter**.Button**(image = **self.**rt\_btn\_img, bd = 0, highlightthickness=0)  **self.**crrCT**.config**(pady = 50)  **self.**crrCT**.grid**(column = 0, row = 2)  **self.**flSe = tkinter**.Button**(image = **self.**wrng\_btn\_img, bd = 0, highlightthickness=0)  **self.**flSe**.config**(pady = 50)  **self.**flSe**.grid**(column = 1, row = 2) |

**34.2 Type-Hint in python**

|  |  |
| --- | --- |
| * Variable type-hint: python has dynamic type property and also Python is strongly typed. But we can also show variable's type as a hint by using "***type-hint***". It don't make it static   #*python has dynamic type property and also Python is strongly typed*  #*But we can also show variable's type as a hint by using "type-hint". It don't make it static*  age = 12  age = "ypung"  #*but if we use "type-hint" the type will be shown by a hint, anywhere of the code*  hour: int  hour = 23  **print**(f"age: {age}, hour: {hour}") |  |
| * Function parameter (input) type-hint: We can use "type-hint" in a for function PARMS/INPUT function declarations. So that if we use that function anwhere in the code, the type will be shown by "hint"   #*we can use "type-hint" in a for function PARMS/INPUT function declarations*  #*so that if we use that function anwhere in the code, the type will be shown by "hint"*  **def** **samp**(a: int):  **print**(a)  #*other codes*  #*other codes*  #*other codes*  #*other codes*  #*other codes*  **samp**(44) |
| * Function output type-hint:  Also we can specify the output of function which will also give us its output type as hint. So that we can use its result properly.   #*we can also use "type-hint" for function OUTPUT in a function declarations*  #*so that if we use the result of the function anwhere in the code, the OUTPUT-type will be shown by "hint"*  **def** **nOsamp**(a: int) -> int:  **print**(a)  **return** a  #*other codes*  #*other codes*  #*other codes*  #*other codes*  #*other codes*  **nOsamp**() |

* Project: Creating a GUI Quiz App from Day – 17.

quiz\_main\_instructor.py

**from** question\_model **import** Question

#*from data import question\_data*

**from** quiz\_brain **import** QuizBrain

**import** ui

**import** html

**import** requests

#*get questions from API*

#*Separate parameters from this URL:*

#*https://opentdb.com/api.php?amount=10&type=boolean*

#*"https://opentdb.com/api.php" is url and "amount=10&type=boolean" are parameters*

parMs = {

    "amount" : 10,

    "type" : "boolean"

}

api\_ques = requests**.get**(url = "https://opentdb.com/api.php", params = parMs)

api\_ques**.raise\_for\_status**()

question\_data = api\_ques**.json**()["results"]

#*Html code conversion: use "html" module and use unescape*

#*question\_text = html.unescape(question["question"])*

question\_bank = []

**for** question **in** question\_data:

    #*Use unescape() for Html entities*

    question\_text = html**.unescape**(question["question"])

    question\_answer = question["correct\_answer"]

    new\_question = **Question**(question\_text, question\_answer)

    question\_bank**.append**(new\_question)

quiz = **QuizBrain**(question\_bank)

#*passing question\_bank to UI module & Generating UI*

quiz\_ui = ui**.quiz\_ui**(quiz)

#*python quiz\_main\_instructor.py*

question\_model.py

**class** Question:

**def** **\_\_init\_\_**(self, q\_text, q\_answer):

**self.**text = q\_text

**self.**answer = q\_answer

quiz\_brain.py

**class** QuizBrain:

**def** **\_\_init\_\_**(self, q\_list):

**self.**question\_number = 0

**self.**score = 0

**self.**question\_list = q\_list

**self.**current\_question = **None**

**def** **still\_has\_questions**(self):

        #*Returns false if questions end*

**return** **self.**question\_number **<** **len**(**self.**question\_list)

**def** **next\_question**(self):

**self.**current\_question = **self.**question\_list[**self.**question\_number]

**self.**question\_number += 1

**return** f"Q.{**self.**question\_number}: {**self.**current\_question**.**text} (True/False): "

        #*user\_answer = input()*

        #*self.check\_answer(user\_answer)*

**def** **check\_answer**(self, user\_answer):

        correct\_answer = **self.**current\_question**.**answer

**if** user\_answer**.lower**() **==** correct\_answer**.lower**():

**self.**score += 1

**return** **True**

            #*print("You got it right!")*

**else**:

**return** **False**

            #*print("That's wrong.")*

**print**(f"Your current score is: {**self.**score}/{**self.**question\_number}")

**print**("\n")

ui.py

**import** tkinter

**from** quiz\_brain **import** QuizBrain

THEME\_COLOR = "#375362"

**class** quiz\_ui:

**def** **\_\_init\_\_**(self, quiz\_brain : QuizBrain): #*type-hint is used*

**self.**quiz = quiz\_brain

**self.**winDow = tkinter**.Tk**()

**self.**winDow**.title**("Quizler")

**self.**winDow**.config**(padx =20, pady = 20, bg= THEME\_COLOR)

**self.**score\_label = tkinter**.Label**(text = "Score 0", fg = "white", bg = THEME\_COLOR)

**self.**score\_label**.grid**(column = 1, row = 0)

**self.**cAnvs = tkinter**.Canvas**(width = 300, height = 250)

**self.**cAnvs**.grid**(column = 0, row = 1, columnspan = 2, pady = 50)

**self.**question\_text = **self.**cAnvs**.create\_text**(

                    150,

                    125,

                    width = 280,

                    text = "Some text will go here",

                    fill = THEME\_COLOR,

                    font = ("Arial", 18, "italic"))

        #*always remember to use keywords: "file"*

        rt\_btn\_img = tkinter**.PhotoImage**(file = "./images/true.png")

        wrng\_btn\_img = tkinter**.PhotoImage**(file = "./images/false.png")

**self.**crrCT = tkinter**.Button**(image = rt\_btn\_img, bd = 0, highlightthickness=0)

**self.**crrCT**.config**(command = **self.**riGht, pady = 50)

**self.**crrCT**.grid**(column = 0, row = 2)

**self.**flSe = tkinter**.Button**(image = wrng\_btn\_img, bd = 0, highlightthickness=0)

**self.**flSe**.config**(command = **self.**woRong, pady = 50)

**self.**flSe**.grid**(column = 1, row = 2)

**self.next\_quiz**()

**self.**winDow**.mainloop**()

**def** **next\_quiz**(self):

**self.**cAnvs**.config**(bg = "white")

**if** **self.**quiz**.still\_has\_questions**(): #*notice "()" is used*

            q\_text = **self.**quiz**.next\_question**()

**self.**score\_label**.config**(text = f"Score : {**self.**quiz**.**score}")

**self.**cAnvs**.itemconfig**(**self.**question\_text, text = q\_text)

**else**:

**self.**cAnvs**.itemconfig**(**self.**question\_text, text = "Reached to the end of the quiz")

**self.**crrCT**.config**(state = "disabled")

**self.**flSe**.config**(state = "disabled")

**def** **riGht**(self):

        is\_right = **self.**quiz**.check\_answer**("True")

**self.give\_feedback**(is\_right)

**def** **woRong**(self):

**self.give\_feedback**(**self.**quiz**.check\_answer**("False"))

**def** **give\_feedback**(self, is\_right):

**if** is\_right:

**self.**cAnvs**.config**(bg = "#4DD637")

**else**:

**self.**cAnvs**.config**(bg = "#B4161B")

**self.**winDow**.after**(1000, **self.**next\_quiz)