

BSC – HGP- Assignment 02

Pictionary Game Specification

1. Assignment Information

Course	BSCO/BSCH
Stage/Year:	3
Module:	HCI & GUI Programming
Semester:	1
Assignment:	2
Date of Issue:	30/10/2024
Assignment Deadline:	12/11/2024
Assignment Weighting:	15% of Module
Assignment Submission:	Via Moodle Only

2. Introduction

In this assignment you will be asked to produce a Pictionary Game Application using PyQt6. This will allow two players to play the game Pictionary whilst in the same room as well as export their drawings should they choose to by create files in a range of different formats.

N.B. You are only awarded marks for what you are asked to do

2.1 Required Features (high-level)

The Pictionary Game is expected to have the following functionality:

- Perform basic file and other operations
- Implement widgets to generate various brush parameters
- Draw lines based on the brush parameters
- Support Gameplay of Pictionary with additional widgets

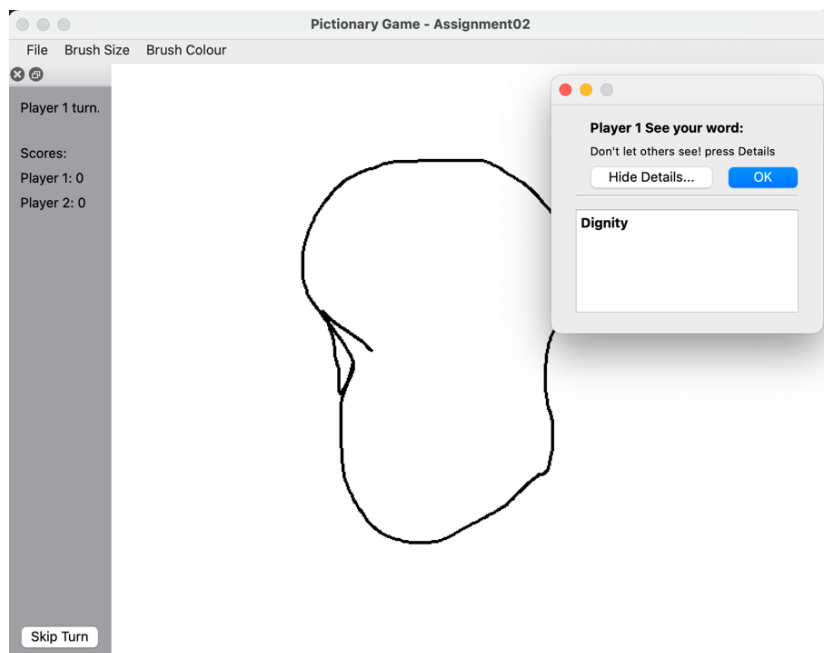


Figure 1 Sample GUI

Please Note the following:

- Figure 1 is a sample GUI included for illustrative purposes only.
- It is missing some features and contains some additional features.
- Although tooltips and a help menu may assist, the app should be intuitive and responsive

3. Features (low-level), Marks and Penalties

The required features are listed here in detail. Failure to implement a feature will result in loss of marks.

There is a degree of flexibility in the method of implementing these features. If you are unclear whether or not your proposed method of implementation is acceptable, please ask the lecturer.

				Marks Breakdown				
Section	Subsection	Feature #	Marks	Present	Functional	Design	Components	
Application (70%)	Application Structure	1	5	2	2	1	Suitable choice of main widget.	
	Menus	2	5	2	2	1	file menu - with options to open, save, clear, exit which execute expected outcomes triggered by selection & shortcuts	
		3	5	2	2	1	help menu - with options for about, help which display information in popups triggered by selection & shortcuts	
	Drawing	4	5	2	2	1	brush_colour - intuitive widget to allow selection and display of a colour e.g. red, black etc	
		5	5	2	2	1	brush_thickness - intuitive widget to allow selection and display of line thickness e.g. 2px, 3px etc.	
		6	5	2	2	1	ability to draw lines on the screen based on the values selected by the brush widgets	
	Gameplay	7	5	2	2	1	suitable widget to display gameplay information to user such as: - scores of each player (award 1 point to guesser and 2 points to drawer on correct guess) - current turn	
		8	5	2	2	1	start game - suitable choice of control that allows the game to start with relevant shortcuts	
		9	5	2	2	1	skip turn - suitable choice of control that allows the current player to skip their turn with relevant shortcuts	
		10	5	2	2	1	display a dialog to show the random word to the current player	
		11	5	2	2	1	mechanism to allow current user to confirm a correct guess, update scores and move to next turn	
	Additional Features	12	5	2	2	1	Additional visual feature e.g. status bar, custom control, etc.	
		13	10	4	4	2	Additional functional feature.	
		Subtotal	70					
Documentation (30%)	Code	12	15				Clearly Comment Code in file - Explanation of method functionality, data structures and underlying logic - Explanation of parameters of methods	Kept precise and clear, complete for all code elements, review provided links for additional tips.
	UI Design Document	13	15				Use template provided. Your submission will only be marked if it includes your completed UI Design Document.	Include screenshots, write clearly under all headings, explain all choices no matter how basic. Explain design choices only, not
			Subtotal	30				
Penalties			Deduction	Error			Reason	
			-30	Non-executable code submitted.			Encourages learner to produce robust code. Reduces marking time.	
			-20	Non-standard libraries used, only standard SDK permitted Wrong compressed file format (only zip and rar accepted)			Ensures equal workload of all students. Reduced Encourages student to distribute resources in easy to read formats. Reduces marking time as additional decompression applications do not need to be installed.	
			-10	Wrong folder structure (see project introduction)			Encourages students to present work in a well structured format. Reduces marking time to determine location and presence of component.	
			-10	Deductions for bugs Standard late penalties			-10% Deduction per bug. As per faculty guidelines.	

Each feature is awarded marks based on










- **Present:** if the feature is present in the application
- **Functional:** if the feature contributes to a well working app, higher marks will be awarded for customization of the function or attributes of the widget
- **Well Designed:** if the feature is incorporated well into the application obeying basic GUI design principles.

N.B. The elements should be clearly reported in the comments in your code and your “UI Design Document.pdf” file.

Any submissions that do not include a completed UI Design Document will receive an automatic 0.

4. Resources to Assist You

studentAnswerTemplate is available on Moodle to download. It contains the following folder and files:









- ▼  studentAnswerTemplateV2
 - ▼  FirstName_LastName_StudentNumber_Ass1
 - ▼  code
 - >  icons
 -  easymode.txt
 -  hardmode.txt
 -  PictionaryGame.py
 -  BSC-HGP - Assignment 02 - Specification.pdf
 -  BSC-HGP - Assignment 02 - UI Design Document.docx

- **Code** – Folder containing the template and txtfiles required for the Assignment
 - **PictionaryGame.py** - the template on which to base the code component of your submission, **edit** this template to add the required functionality as per above
 - **easymode.txt** – text file containing random words for the game, read in from the local directory using the `getList()` method and sending in the string “easy”
 - **hardmode.txt** – text file containing random words for a more difficult version of the game, read in from the local directory using the `getList()` method and sending in “hard”
 - **icons** – folder containing relevant icons for the template **edit** add any additional icons used for your Game
- **BSC-HGP - Assignment 02 - UI Design DocumentV1.docx** – **edit** to explain design choices and highlight additional features
- **BSC-HGP - Assignment 02 - Specification.pdf** – relevant details on the assignment, **do not edit** this document

Import all files from the **Code** folder into your PyCharm project for Assignment02 and edit the PictionaryGame.py template to include the functionality as per section 3.

5. Submission

Your final submission should be structured as below

- ▼  studentAnswerTemplateV2
 - ▼  FirstName_LastName_StudentNumber_Ass1
 - ▼  code
 - >  icons
 -  easymode.txt
 -  hardmode.txt
 -  PictionaryGame.py
 -  BSC-HGP - Assignment 02 - UI Design Document.pdf

- **Rename** FirstName_LastName_StudentNumber_Ass2 to your details
- **Compression** folder to zip or rar
- **Submit** to Moodle

Don't forget to include relevant assets (such as stylesheets, icons etc) in your submission.

Email submissions will not be accepted, you must submit via the Submission Point on Moodle.

6. Steps to Complete This Assignment

1. Drag all three files from the Code folder into your PyCharm project for Assignment 02.
Start editing *PictionaryGame.py*
2. Follow the video *PyQt5 Creating Paint Application In 40 Minutes*
<https://www.youtube.com/watch?v=qEgyGyVA1ZQ> to ensure a complete understanding
3. Add intuitive Widgets to improve and add to the existing ones using the tutorial resources listed below, and widgets described below. Be sure to implement all the features in section 3. *Features(low-level), Marks & Penalties*
 - <https://zetcode.com/pyqt6/menustoolbars/>
 - Add a toolbar using the addToolBar() method.
 - Add actions you have already made to the toolbar using the addAction() method
 - Add other widgets to the toolbar using the addWidget() method e.g. QPushButton (this could cause a QColorDialog to popup), QRadioButton, QComboBox, QDial
 - Will help you with
 - Status Bar
 - Simple Menu
 - Sub Menu
 - Check Menu
 - Context Menu
 - Toolbar
 - <http://zetcode.com/pyqt6/dialogs/>
 - Will help you with
 - File Dialogs
 - Help Menu Popups
 - <http://zetcode.com/pyqt6/painting/>
 - Will help you with
 - Colours
 - Points
 - Lines
 - QPen
 - ComboBox containing images
 - <https://stackoverflow.com/questions/21016945/pyqt-images-in-combobox-items>
 - <http://zetcode.com/pyqt6/customwidgets/>
 - One option for an additional feature
 - Other features are possible.
4. Add icons to your menu from here 32x32 pngs from - <https://www.flaticon.com/>
5. Explore the use of the following widgets:
 - CheckBox
 - QRadioButton
 - https://www.tutorialspoint.com/pyqt/pyqt_qradiobutton_widget.htm
 - QPushButton
 - QTabWidget
 - QTableWidget
 - QScrollBar
 - QProgressBar
 - QDateTimeEdit
 - QSlider
 - QDial
 - <https://stackoverflow.com/questions/18486501/pyqt4-qt-set-orientation-of-qdial-minimum-value-at-the-top>

- QGroupBox
- QCalendarWidget
- QLabel
- QDateEdit
- QComboBox with images
 - <https://stackoverflow.com/questions/21016945/pyqt-images-in-combobox-items>

Further information on these widgets can be found here:

- <https://doc.qt.io/qt-6/gallery.html>

7. Documentation

1. Qt Documentation
 - Widgets: <https://doc.qt.io/qt-6/qwidget.html>
 - Modules: <https://doc.qt.io/qt-6/qtmodules.html>
2. PyQt Documentation
 - <https://www.riverbankcomputing.com/static/Docs/PyQt6/api/qtwidgets/qtwidgets-module.html>
3. Documenting Your code
 - <https://realpython.com/documenting-python-code/> (you can just use # and a good explanation!)