

레포트 #2 - 보드게임 + API 활용 예제

개요

- 1) 이 프로그램은 '가위바위보-하나빼기일' 게임을 gui 버전으로 구현하는 것을 목표로 한다.
- 2) 게임을 시작할 때, 유저는 '주먹', '가위', '보', 세 버튼 중 둘을 눌러서 무엇을 낼지 선택을 한다. 둘을 선택하면 유저와 컴퓨터가 선택한 두 개가 나온다. 이때 그 둘 중 하나를 고르면 승/패 결과가 나오도록 코드를 만들었다.
- 3) 게임은 인간 유저와 컴퓨터 사이에서 이루어진다. 인간 유저가 컴퓨터를 이길 때마다 포켓몬이 나오면서 축하 팝업이 나오며, 유저가 지면 'ㅠㅠ. 졌습니다.' 라는 문구가 나오도록 하였다. 비겼을 때도, '비겼습니다' 같은 문구가 나온다.
- 4) 게임의 긴장감을 더하기 위해서 유저와 컴퓨터의 점수를 각각 나타내는 점수판을 만들었다. 한 판을 누군가 이길 때마다 점수가 1씩 올라간다.
- 5) 게임 한 판이 끝나면 '다시하기' 버튼을 누르면 된다. 게임이 renewal된다.

코드 분석과 구현 결과

-Settings

#첫 부분

위의 부분은 Python의 **tkinter 모듈**을 활용하여 그래픽 사용자 인터페이스(GUI)를 구성하는 기본 환경을 제공한다. 또한 **random, json, urllib.request** 모듈을 통합하여, 무작위 데이터 생성 및 웹 기반 API로부터의 데이터 요청 기능을 구현할 수 있도록 설계되었다. 아울러 **base64 모듈**을 활용하여 이미지 및 데이터를 인코딩·디코딩하는 기능을 지원함으로써, 다양한 데이터 처리 및 시각화 요구를 충족할 수 있도록 구성되어 있다.

Def_init_(self, master);, 함수 정의를 통해서 게임 창의 이름, 크기, 특성(창 크기 조절 가능 유무)등을 설정했다.

#Game options and state variables

Self.options이라는 딕셔너리를 생성했다. '바위', '가위', '보'에 대응되는 value가 설정되어 있다. 컴퓨터의 선택과 유저의 선택 항목을 비워 놓았고, 컴퓨터의 점수와 유저의 점수를 0으로 설정하였다.

#Setup fonts

Default, info, status, result, label에 대한 글자의 글꼴과 사이즈, 굵기 등이 설정되어 있다.

#---UI Widgets---

"첫 번째 손을 선택하세요,"라는 문구를 화면에 놓기 위해서, self.info_label이라는 tk.label의 결과값으로 설정하고 pack()을 사용했다. 같은 방법으로 가위, 바위, 보 버튼을 배치할 프레임 설정하였다. For 반복문을 이용해서 버튼들을 저장할 self.buttons라는 딕셔너리를 만들었다. 이때 for문을 통해서 "바위", "가위", "보" 각각에 대해 버튼 생성을 반복한 뒤, 배치한다.

-Code for strategic choices

#---Strategic Choice Section--- 전략 선택 단계를 위한 프레임을 생성하고 pack()을 사용해서 배치하였다.

#Player's side

"하나빼기" 단계에 플레이어가 선택해야 할 두 가지 옵션과 그것의 프레임이 창의 왼쪽에 위치하도록 한다. 전과 같은 방법으로 "나의 패"라는 라벨을 설정하고 pack()을 이용해서 창의 특정 위치시킨다. (최종 선택 버튼들을 저장하기 위한 리스트<참고>)

#Computer's side

컴퓨터쪽의 전략 선택 버튼도 플레이어와 비슷한 방법으로 배치된다. 다만 다른 점이 있다면, 플레이어가 세 개의 전략적 옵션을 고르기 전, 컴퓨터의 전략적 선택 프레임 안에는 '?'표시가 있어야 한다는 것이다. 그래서 컴퓨터의 전략 선택을 각각 hand1과 hand2일 때, 두 개의 text를 '?'로 만들었다.

#---Result Section---

플레이어가 남은 두 가지 전략 중 하나를 선택하면 결과가 나온다. 결과 표시용 프레임을 설정하고 창에 넣는 작업을 pack()을 사용해서 하였다. 최종 선택된 패, '다시하기'를 나타내는 label에 대한 설정과 배치, 그리고 라벨이 들어갈 프레임(그리고 버튼)에 관한 코드들이 나온다. Def get_score_text(self): 각자의 점수를 '-'를 사이에 두고 나타낸다.

Def make_initial_choice(self,choice): 는 #Game options and state variables부분에서 만든 self.player_choices라는 리스트의 길이와 if문을 활용해서 세 가지 전략 중 한 가지를 선택

택하는 과정을 규정하였다. If-elif문을 활용해서 list의 길이가 1일 때와 2일 때, 다른 동작이 수행되도록 하였다.

Def setup_final_choice_phase(self): <‘하나빼기’ 단계에서 플레이어의 동작 과정을 규정함> 세 가지 전략 중 하나를 고르면 ‘하나빼기’ 단계에 접어들면서 컴퓨터의 전략 선택 프레임에 두 가지의 전략이 각 프레임에 위치하게 된다. 그리고 플레이어가 고를 두 개의 전략 버튼이 생긴다. 이 함수 정의에서는 for문을 이용해서 self.player_choices에 있는 choice 각각을 위한 버튼이 반복 생성된다. 마지막으로 btn??

Def get_computer_ai_choice(self): 는 컴퓨터가 ‘ai’로 플레이어의 전략을 예측하여 전략을 선택하는 함수이다. 이때, max(player_outcomes_vs_c1)는 컴퓨터가 c1을 선택할 때, 플레이어를 이길 수 있는 확률을 계산한다. Play_outcomes_vs_c1은 발생할 수 있는 모든 상황들에 대한 결과들(-1,0,-1)이 나열되어 있는 리스트이다. (이는 def get_outcome에서 정의된 바가 있다.) 컴퓨터가 ai로 확률을 계산해서 그런지 게임을 진행한 결과 플레이어(보고서 작성자)가 컴퓨터에게 15:1로 지기도 했다.

-api 활용

Def show_win_popup(self): 플레이어가 승리하였을 때, 나오는 팝업창에 관한 함수이다. 팝업창이 나오는 과정과 경우는 try-except구문으로 나타내었다.

Try 부분: 1부터 898의 숫자를 무작위로 구한 뒤, 그 숫자에 대응되는 포켓몬 데이터를 가져온다. 포켓몬 api에서 Json 데이터를 받아와서 이미지 데이터를 객체(image_label)로 만든 뒤, 라벨을 팝업에 표시하는 과정을 담고 있다.

Except부분: 오류가 발생했을 경우 에러 메시지를 출력하도록 하고 있다. Except부분이 끝나면 팝업창에 나올 문구 표시, 닫기 버튼에 대한 코드가 있다,

Def play_round(self, player_final_choice_korean): 은 플레이어의 마지막 결정과 컴퓨터의 마지막 결정 과정 그리고 승/패에 따른 결과에 관한 코드이다. 컴퓨터와 플레이어가 마지막 결정을 한 순간, 그것에 따른 결과가 정해지고 이기는 쪽에 점수 1이 추가가 된다. #Game options and state variables부분에서 각자의 점수를 처음에는 0으로 설정해놓고, 게임이 끝날 때마다 이기는 쪽이 1점을 얻는 방식이다. 중복하여 얻은 점수는 화면에 vs를 사이에 두고 표시된다.

Def reset_game(self): 게임을 reset하는 과정을 규정하는 함수이다. ‘다시하기’ 버튼을 비활성화하는 것으로 마친다.

