# Example 1. Guess-the-day ver1

|  |  |
| --- | --- |
| ver0 | original description |
| ver1 | finding class, attribute and method candidates |
| ver2 | Refining the public interface, refining attributes, methods, classes |

## Original description

Select-the-day, the program shows that day’s weekday and three other randomly picked week days, asks a user to select which day is today, checks the answer and saves the information to data storage accessible to the personnel of the service home. If the user picked the wrong day, the program asks again giving a gentle hint e.g. today is the first day of the week. And if the user guesses wrong the second time, the program shows a nice greeting and sets the game as FAILED to the data storage. If the user answered correctly the second time the game is saved as FOLLOW, and if the answer was right the first time, the game is saved as PASSED. The personnel checks the saved games to verify if the resident has problems in keeping up with the time.

## 1st iteration, check nouns

**Select-the-day, the program shows that day’s weekday and three other randomly picked week days, asks a user to select which day is today, checks the answer and saves the information to data storage accessible to the personnel of the service home. If the user picked the wrong day, the program asks again giving a gentle hint e.g. today is the first day of the week. And if the user** **guesses wrong the second time, the program shows a nice greeting and sets the game as FAILED to the data storage. If the user answered correctly the second time the game is saved as FOLLOW, and if the answer was right the first time, the game is saved as PASSED. The personnel checks the saved games to verify if the resident has problems in keeping up with the time.**

|  |  |  |  |
| --- | --- | --- | --- |
| noun | basic form, synonym, duplicate value, notes | value or collections | possible class, object, attribute |
| **Guess-the-day** |  | one value, name of the game, title | **Possible attribute:**  game’s title |
| **game** |  | object, one run of the game inside the program | **Possible class:**  knows the rules of the game and keeps track of the game’s data |
| **weekdays** | weekday | Object | **Possible object:**  Stores three randomly selected weekdays |
| **user** |  | object | **Possible class:** |
| **today** |  | object | **Possible attribute:**  Stores today’s weekday |
| **days** |  | object | **Possible attribute:**  Stores days user can select from |
| **attempts** |  | Type integer | **Possible attribute:** Keeps count of user attempts |
| **answer** |  | Type boolean | **Possible attribute:**  Keeps value of user’s answers |
| **ID** |  | integer | **Possible attribute:**  Identifies game with iD number. ID is used to tell which game is selected of gamelist. |

## 2nd iteration, check verbs

**Select-the-day, the program shows that day’s weekday and three other randomly picked week days, asks a user to select which day is today, checks the answer and saves the information to data storage accessible to the personnel of the service home. If the user picked the wrong day, the program asks again giving a gentle hint e.g. today is the first day of the week. And if the user** **guesses wrong the second time, the program shows a nice greeting and sets the game as FAILED to the data storage. If the user answered correctly the second time the game is saved as FOLLOW, and if the answer was right the first time, the game is saved as PASSED. The personnel checks the saved games to verify if the resident has problems in keeping up with the time.**

|  |  |  |
| --- | --- | --- |
| verb | subject – object (who does, who is the target) | possible action, function, method |
| **pick** | Game’s action | **Possible method:**  Randomly select three other weekdays |
| **select** | User’s action | **Possible method:**  Select which day is today |
| **check** | user’s action | **Possible method:**  Check is user’s answer correct |
| **save** | Games’s action | **Possible method:**  Saves user’s answer to be accessed by service home personnel |
| **verify** | Game’s action | **Possible method:**  Personnel check user answers in later time |
| **ask** | Game’s action | **Possible method:**  Ask if wanted to retrieve recently created answers + show answers |
| **guess** | User’s action | **Possible method:**  Allow user to make guess of todays’ weekday |
| **show** | Game’s action | **Possible method:**  Show 4 weekdays |
| **quit** | Games’s action | **Possible method:**  Quits the game |
| **restart** | Games’s action | **Possible method:**  Restart the game |

## 3rd iteration

@ver2 discription

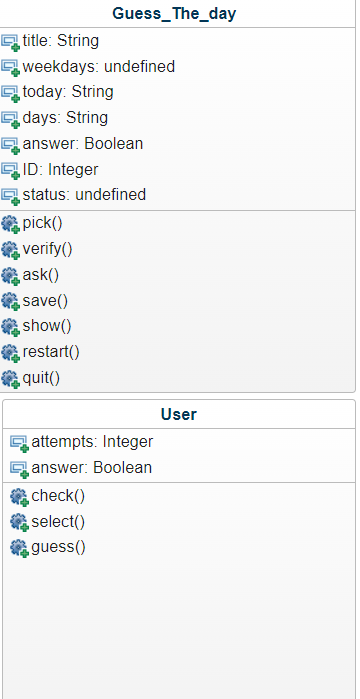
Guess The Day game asks user to select from 4 options which weekday is today. Game will save user answer as true or false. If first attempt fails, user is given gentle hint and allowed to try again. If second attempt fails. User gets nice hint and game ends. If user answers correctly on one of the times, user gets message “passed”. After the game medical personnel can check users’ answers.

### Description

@Ver 1, @ver2

1. class guess the day
   1. Attributes
      1. Title
         1. Stores title of the game
      2. Weekdays
         1. Stores all weekdays
      3. Today
         1. Stores today’s weekday
      4. days
         1. Stores days user can select from
      5. ID
         1. Identifies game with ID number. ID is used to tell which game is selected from games list
      6. \_status
         1. Stores three states which are used to keep track of current state of the game
      7. Hints
         1. Stores hints user are given in different states of the game
   2. Methods
      1. Pick
         1. Randomly select three wrong weekdays
      2. Verify
         1. Personnel check user answer in later time
      3. Ask
         1. Ask if wanted to retrieve recently created answers + show answers
      4. Save
         1. Saves user’s answer to be accessed by service home personnel
      5. Show
         1. Show 4 weekdays to choose one
      6. Restart
         1. Restarts the game
      7. Quit
         1. quits the game
2. class user
   1. Attributes
      1. Attempts
         1. keeps count how many times user has answered
      2. Answer
         1. Stores user’s answers
   2. Methods
      1. Check
         1. Check user answer whether it is wrong or right
      2. Select
         1. select which weekday is today
      3. guess
         1. Allow user to make guess of todays’ weekday
   3. Reserve topic
   4. Reserve topic

### Generated UML class diagram (<https://app.genmymodel.com/>) @ver1, @ver2



### Code generated from the UML diagram @ver1. @ver2

**class Guess\_The\_day(object):**

**def \_\_init\_\_(self):**

**self.title = ""**

**self.weekdays =**

**self.today = ""**

**self.days = ""**

**self.answer = False**

**self.ID = 0**

**self.status =**

**# Start of user code -> properties/constructors for Guess\_The\_day c**

**# End of user code**

**def pick(self):**

**# Start of user code protected zone for pick function body**

**raise NotImplementedError**

**# End of user code**

**def verify(self):**

**# Start of user code protected zone for verify function body**

**raise NotImplementedError**

**# End of user code**

**def ask(self):**

**# Start of user code protected zone for ask function body**

**raise NotImplementedError**

**# End of user code**

**def save(self):**

**# Start of user code protected zone for save function body**

**raise NotImplementedError**

**# End of user code**

**def show(self):**

**# Start of user code protected zone for show function body**

**raise NotImplementedError**

**# End of user code**

**def restart(self):**

**# Start of user code protected zone for restart function body**

**raise NotImplementedError**

**# End of user code**

**def quit(self):**

**# Start of user code protected zone for quit function body**

**raise NotImplementedError**

**# End of user code**

**# Start of user code -> methods for Guess\_The\_day class**

**# End of user code**

**class User(object):**

**def \_\_init\_\_(self):**

**self.attempts = 0**

**self.answer = False**

**# Start of user code -> properties/constructors for User class**

**# End of user code**

**def check(self):**

**# Start of user code protected zone for check function body**

**raise NotImplementedError**

**# End of user code**

**def select(self):**

**# Start of user code protected zone for select function body**

**raise NotImplementedError**

**# End of user code**

**def guess(self):**

**# Start of user code protected zone for guess function body**

**raise NotImplementedError**

**#** End of user code

Completed code @ver 1, @ver2

**# File: guess\_the\_day.py**

**# Author(s): Pavel Kaljunen, Sebastian Sopola, Uras Ayanoglu, Jerry Karkainen**

**# Description: This game is memory game. It is played to check if user gets day's weekday correct.**

**# Import necessary libaries**

**# This class handles game interaction**

**class Guess\_The\_day:**

**# Establishing parameters**

**def \_\_init\_\_(self):**

**self.title = ""**

**self.weekdays = ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"]**

**self.today = ""**

**self.days = ""**

**self.answer = bool**

**self.ID = 3**

**self.\_status = ("passed", "failed", "follow")**

**self.hints = ["Unfortunately this time you didn't guess it right.", "Today is first day of the week", "Today is second day of the week", "Today is third day of the week"]**

**# Start of user code -> properties/constructors for Guess\_The\_day class**

**# End of user code**

**def pick(self):**

**# randomly generate three wrong weekdays**

**pass**

**def verify(self):**

**# Personnel check user answers in later time**

**pass**

**def ask(self):**

**# Ask if wanted to retrieve recently created answers + show answers**

**pass**

**def save(self):**

**# Saves user s answer to be accessed by service home personnel**

**pass**

**def show(self):**

**# Show 4 weekdays to choose one**

**pass**

**def restart(self):**

**# User can restart the game**

**pass**

**def quit(self):**

**# User can quit game**

**pass**

**# This class handles user interaction**

**class User:**

**# Establishing parameters**

**def \_\_init\_\_(self):**

**self.attempts = 0**

**self.answer = ""**

**def check(self):**

**# Check user answer**

**pass**

**def select(self):**

**# select which weekday is today**

**pass**

**def guess(self):**

**# Start of user code protected zone for guess function body**

**pass**

**# End of user code**