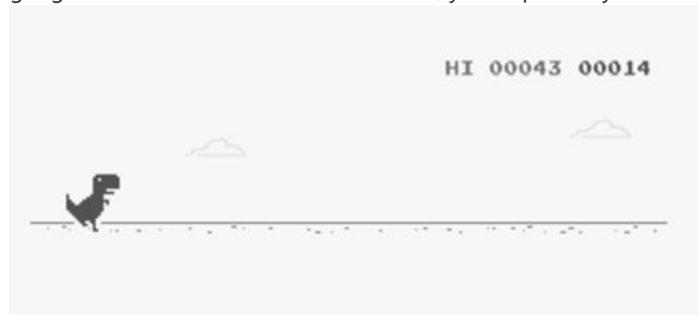


Bot for Google Chrome's Dinosaur Game

If you've ever tried to access google chrome while the internet is down, you've probably seen a screen similar to this:



T-Rex Run is a simple game. The user plays as the dinosaur, using the space bar to jump over incoming cacti and birds. It's an easy game to play, and python makes it even easier.

Automate the Game

In order to automate the game, there are two user inputs that we have to consider:

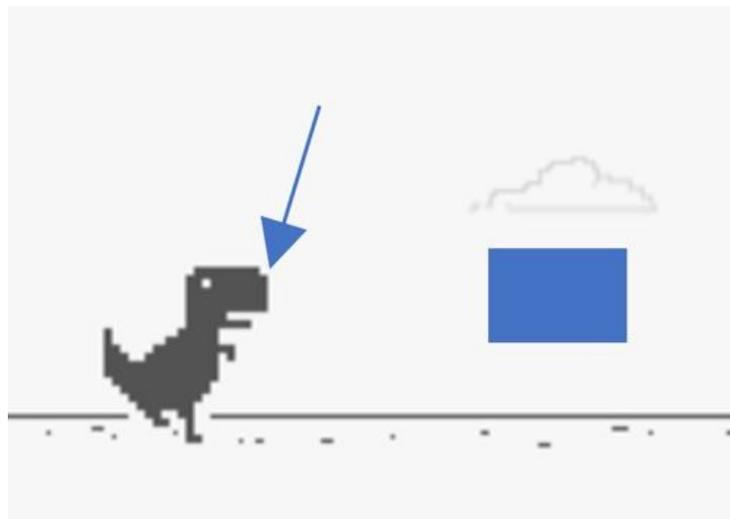
1. Clicking the center to start/restart the game
2. Pressing and unpressing the space bar to jump

To accomplish these tasks, three python modules are used:

1. [NumPy](#) - for summing pixel values
2. [Pillow](#) - for image processing

Conceptual Overview

The goal is to make the program able to analyze the area in front of the dinosaur and jump when something is in his way. The following image shows the two places that the program is using to do this:



The blue rectangle is a reference to the area of the screen in front of the dinosaur that the bot is constantly running calculations on. As the game runs, the area in front of the dinosaur changes as obstacles approach. With Pillow, we're able to calculate the sum of the different pixel values specifically in the blue area in front of the dinosaur.

When nothing is in front of the dinosaur, the value of the sum of these pixels represented by the blue rectangle is 1,447. As a cactus or bird approaches the blue area, the value of the pixels changes because the dark color of the birds and cacti have a different pixel value than the white pixels in the background. Once the program calculates that there is a difference, it knows to then execute a "jump" because something is in the way.

The "box" variable in the [python script](#) is the code representation of the blue rectangle. Its coordinates are created in reference to the front of the dinosaur where the blue arrow is pointing to in the image above.

The blue arrow points to the pixel location of the "dino" variable in the *Coordinates* class below.

```
class Coordinates():
    # Pixel location on screen of restart button
    replay_button = (480, 580)
    # Pixel location just in front of dino
    dino = (220, 585)
```

It's important to note that this variable will be different depending on the location of the game on your screen. PyAutoGUI executes clicks based on the (x, y) coordinate plane of your screen, so in order for the script to work properly, the chosen values for the *dino* and *replay_button* tuples must be from those appropriate locations on your screen. The *replay_button* tuple is the (x, y) pixel location of the replay button shown at the end of a run.

And that's pretty much it! The game will continue to run and the program will continue to jump over cacti and birds until the script is terminated.

Code Walk-Through

The [python code](#) has inline comments, but here's a step-by-step walk-through of how the script works

1. The bot is activated with the following command from the terminal:

```
python dino_bot.py
```

2. PyAutoGUI clicks on the pixel location of the restart button to begin a new game via the `restart_game()` function
3. A count variable is initiated to keep track of total jumps that the bot executes
4. While the bot is running, the `image_grab()` function calculates the pixel value sum for the area in front of the dinosaur
5. If the value of the 40x30 pixel grid in front of the dinosaur is 1447, do nothing.
6. If the value of the 40x30 pixel grid in front of the dinosaur is anything other than 1447, jump and raise the jump count by one. This feature was added to help prove that the bot is executing the jumps (not a human) in the gif of the program running at the top.

And that's all! Overall, this was a fun and easy way of showing how powerful just a few lines of python can be.

Author

- Jimmy Dudley - [website](#) - [email](#) - [github](#)