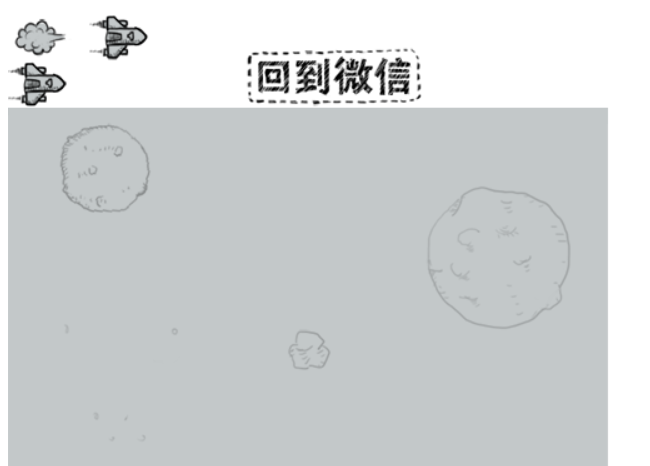
The repository Kill-Console/PythonShootGame contains 3 files:

* A **README** file with the overall description about the project, that consists in a simple shoot game written in Python.
* An **mainGame.py** file with the initialization and main loop of the game.
* A **gameRole.py** file with the class of the game role.

Plus, the repository contains a folder **resources**, containing the graphical (images) and media files used on the game.



Source: GitHub Kill-Console/PythonShootGame

The game consists on a 2D platform aircraft shooter game, like the classic Atari - River Raid. Enemies go down on the screen and the user has to shoot (vertically) to destroy them while avoid being hit. What is most impressive on this project is the graphic interface that the developers were able to create in Phyton using a small (but very well O.O structured) code and a simple image library.

What was clear to me after analyzing the project:

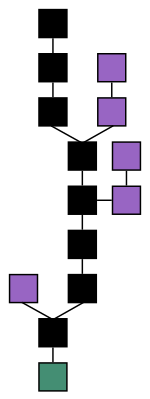
1. The image files that the author provided made it easier to understand the mechanics of the game.
2. I was able to understand how the game respond to users events – like a keypad press – on the mainGame code.
3. Overall, the code is very well structured, clean and object-oriented, which makes it easier to read and analyze it.

What was not clear to me:

1. It was difficult to obtain information about the project, since it is not about a famous IT concept and algorithms (like, for instance, the Neural Network / LSTM models on the previous project). The README file is very poor in terms of documentation, and additional documentation provided was in Chinese only.
2. Trying to understand the code was also difficult due my lack of knowledge on Phyton and by the fact that the comments on the code were in Chinese.
3. How the developers were able to convert on the code the library images into motion pictures in the game.

The repository Carlos-Zen/blockchain-python contains a dozen of Python files with algorithms that implement a blockchain algorithm in Python only for study.

A blockchain is a growing list of records - called blocks - that are linked using cryptography. Each block contains a cryptographic hash of the previous block, a timestamp and transaction data. Is generally represented as a Merkle tree, like the image below. The main chain (black) consists of the longest series of blocks from the genesis block (green) to the current block. Orphan blocks (purple) exist outside of the main chain.



Blockchain formation. Source: Wikipedia

Blockchain solution is considered one of the main technological advances from Bitcoin.

The design has inspired other applications, such as the proposed Blockchain-python (this project) that implements simple blockchain and transactions using mining, transaction, communication between nodes and file persistence of blocks and transactions. In order to simplify the solution, the communication between nodes was implemented via RCP protocol (based on http), rather than P2P network. The author admits however that the solution is not ready yet, because they were still not able to implement the verification based on cryptography.

These are the most relevant files at the repository:

* **account.py:** account structure.
* **block.py:** blocking data structure (for the blockchain algorithm).
* **database.py:** database information for the transactions.
* **miner.py:** a simplified mining algorithm (implement only rewards) based on sha256 hash.
* **model.py:** implements the class model importing a JSON library.
* **node.py:** the node data structure for the blockchain algorithm. Implements graph algorithms basic properties such as get node, check node and add node.
* **rpc.py:** implementation of the RPC protocol, a data exchange protocol that allows a client (for example: a blockchain explorer) to communicate to a server (your local blockchain node) by issuing commands and listening to responses.
* **transaction.py:** implements the class transaction, that uses most of the previous mentioned structures.

What was clear to me after analyzing the project:

1. Code was clear and well-structured according object-oriented concepts.
2. It was easy to find more information and read more about blockchain algorithms, mining and hashing solutions.
3. I have a reasonable knowledge about graph data structure and algorithms, which helped-me me to understand the code (specially in node.py and block.py).

What was not clear to me:

1. Although I was able to reasonable understand the isolated codes, I found it hard to understand the flow of the program, and how the different algorithms (in different files) connects. The fact that the solution is still incomplete made it even more difficult.
2. The role of the Class Model on this implementation. The lack of comments on the code worsen the problem.
3. Integration between Java (JSON) code and Phyton code.