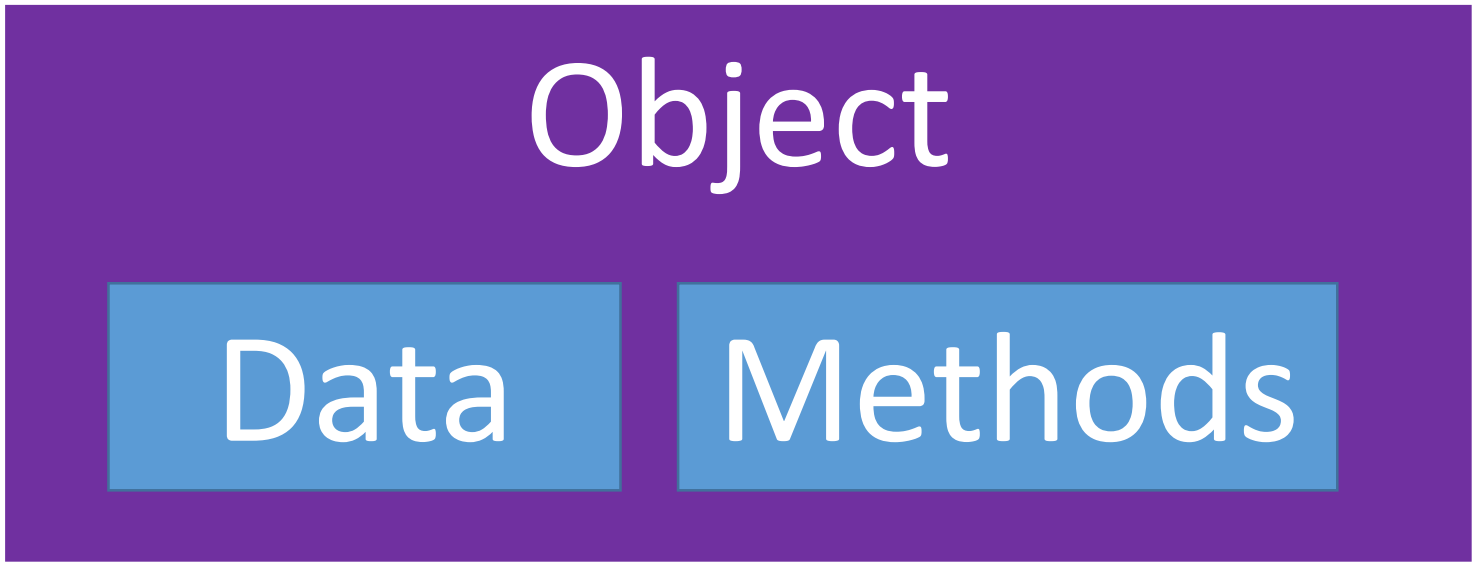
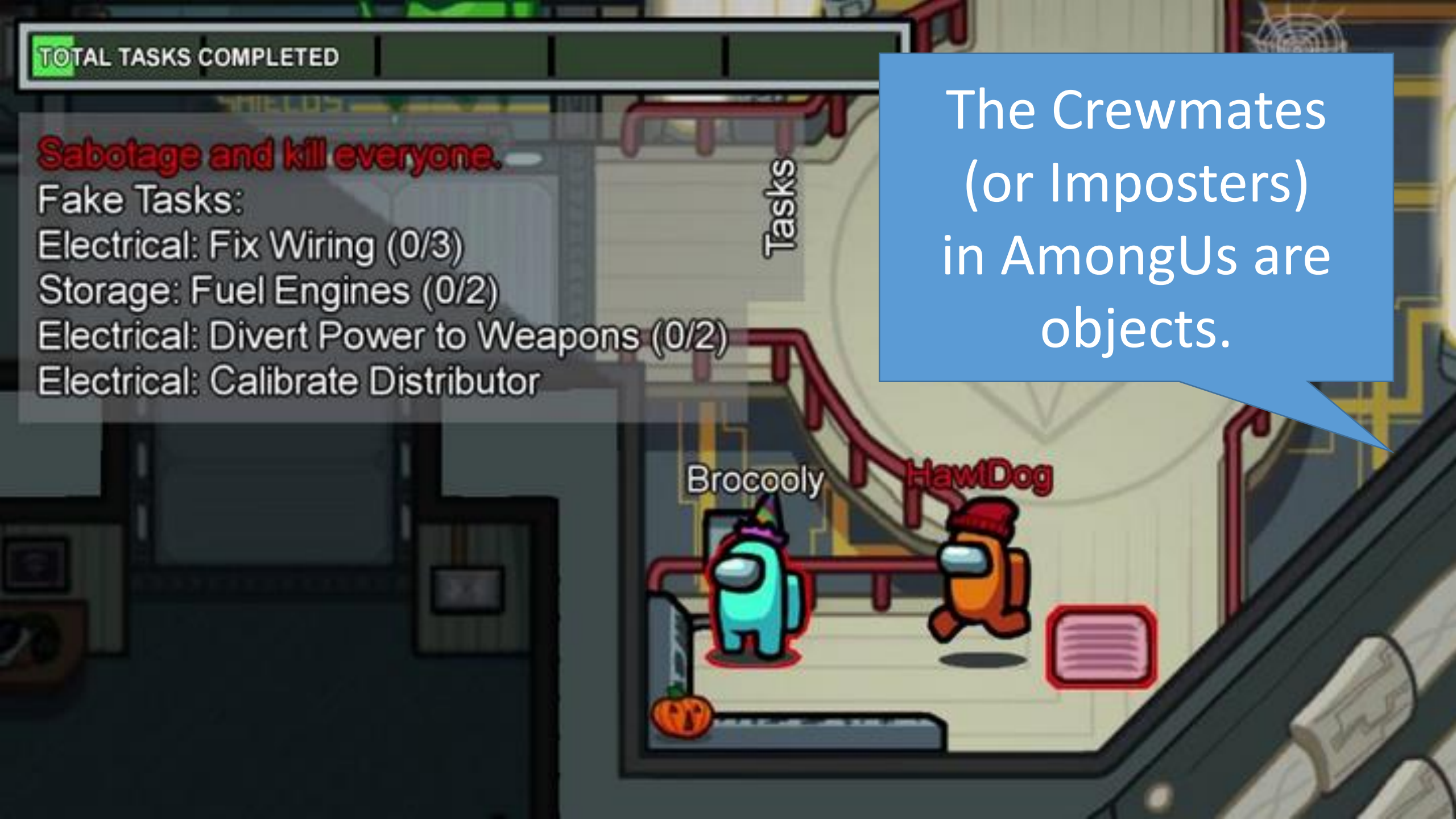
# Finding Objects

Objects have a very general vague definition.

That's because it covers a lot of really different things.

Students don't like this definition, sorry, but it is an accurate one.



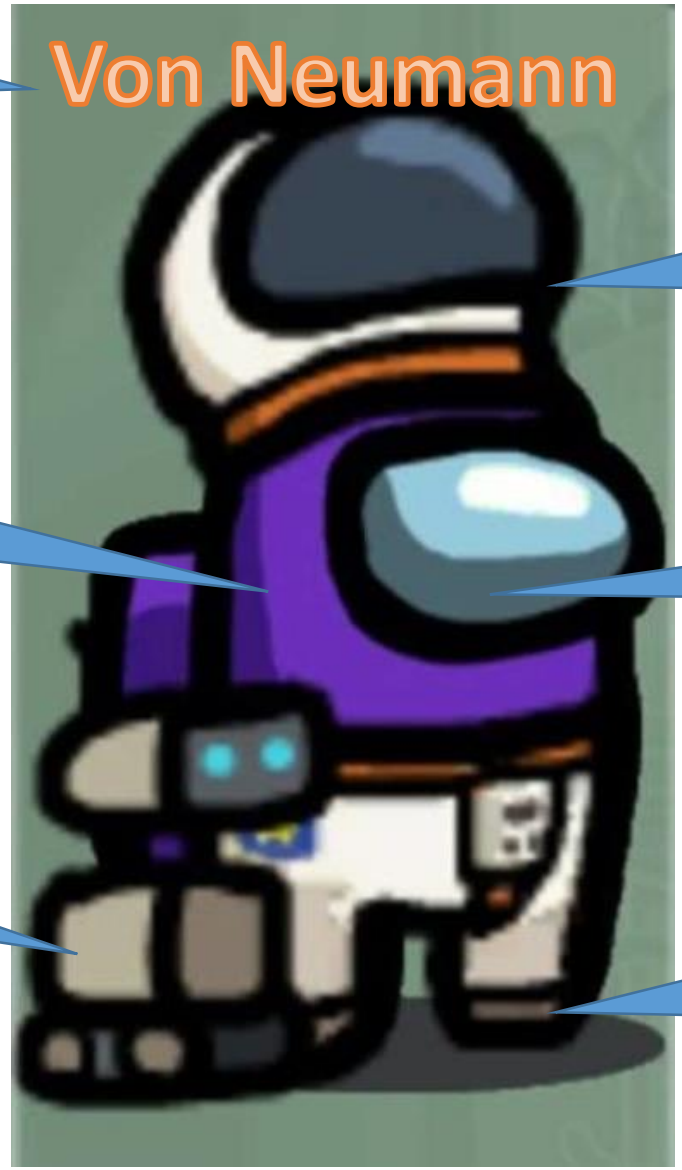

TOTAL TASKS COMPLETED
Sabotage and kill everyone.
Fake Tasks:
Electrical: Fix Wiring (0/3)
Storage: Fuel Engines (0/2)
Electrical: Divert Power to Weapons (0/2)
Electrical: Calibrate Distributor
Tasks
The Crewmates (or Imposters) in AmongUs are objects.
Brocooly
HawlDog

The option menus let you see some of the data stored.

Selecting an option and applying it to a crewmate is a method.

Data

Name



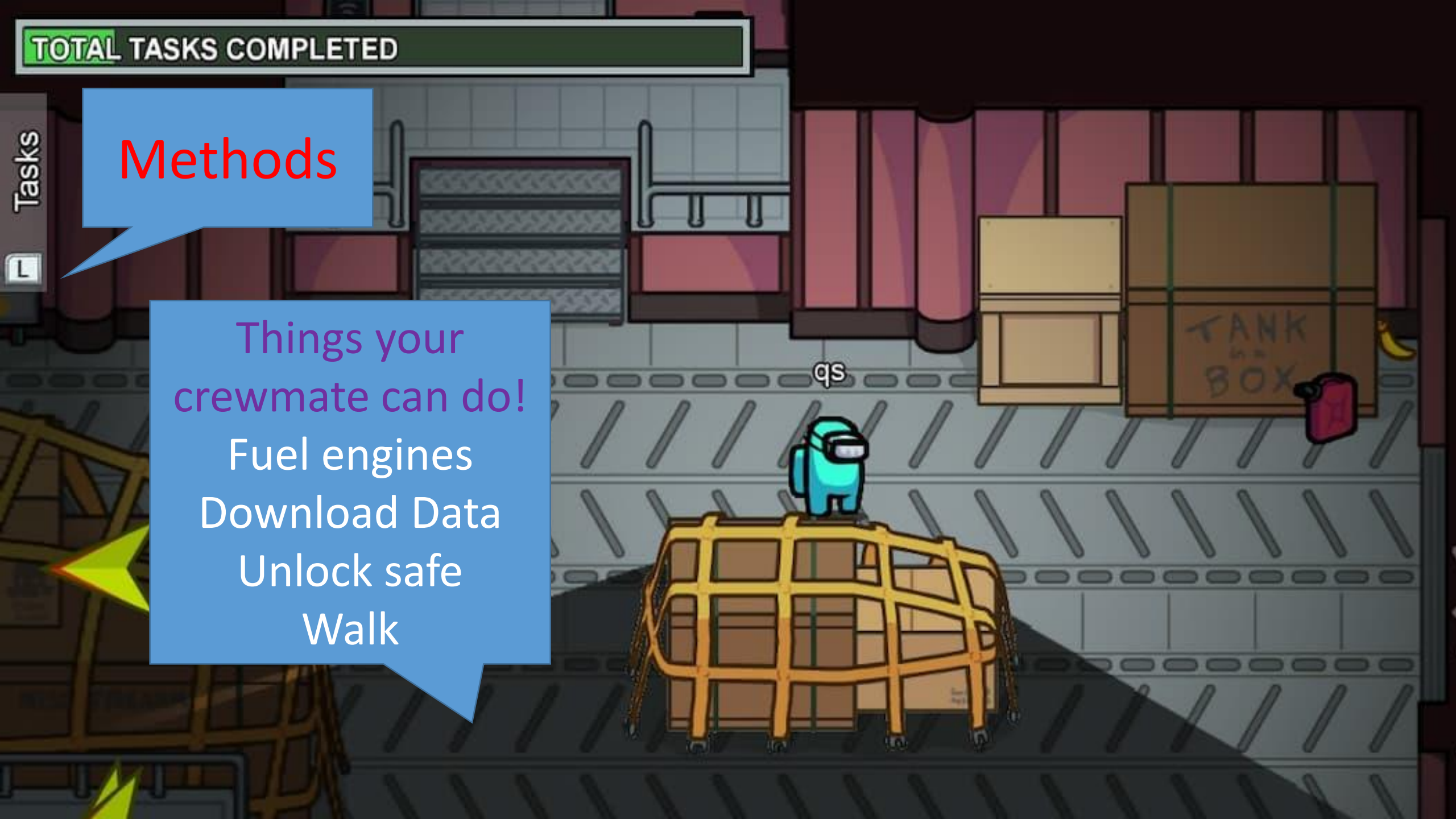

Hat

Colour

IsImposter

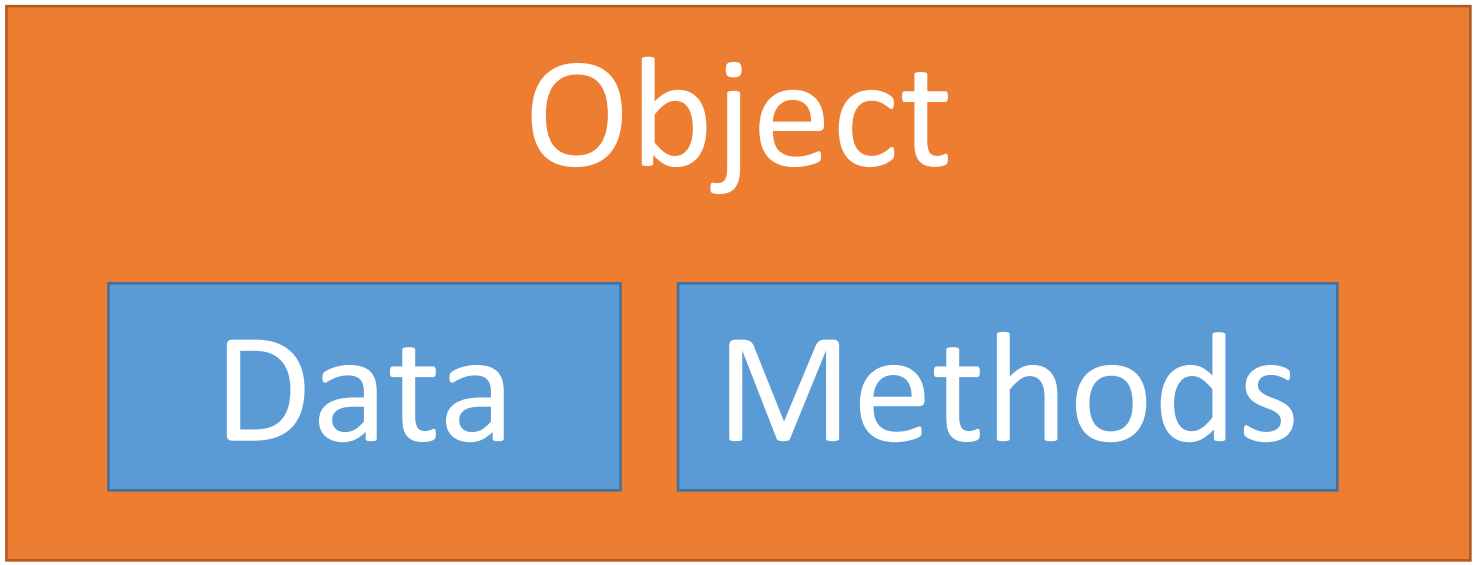
Pet

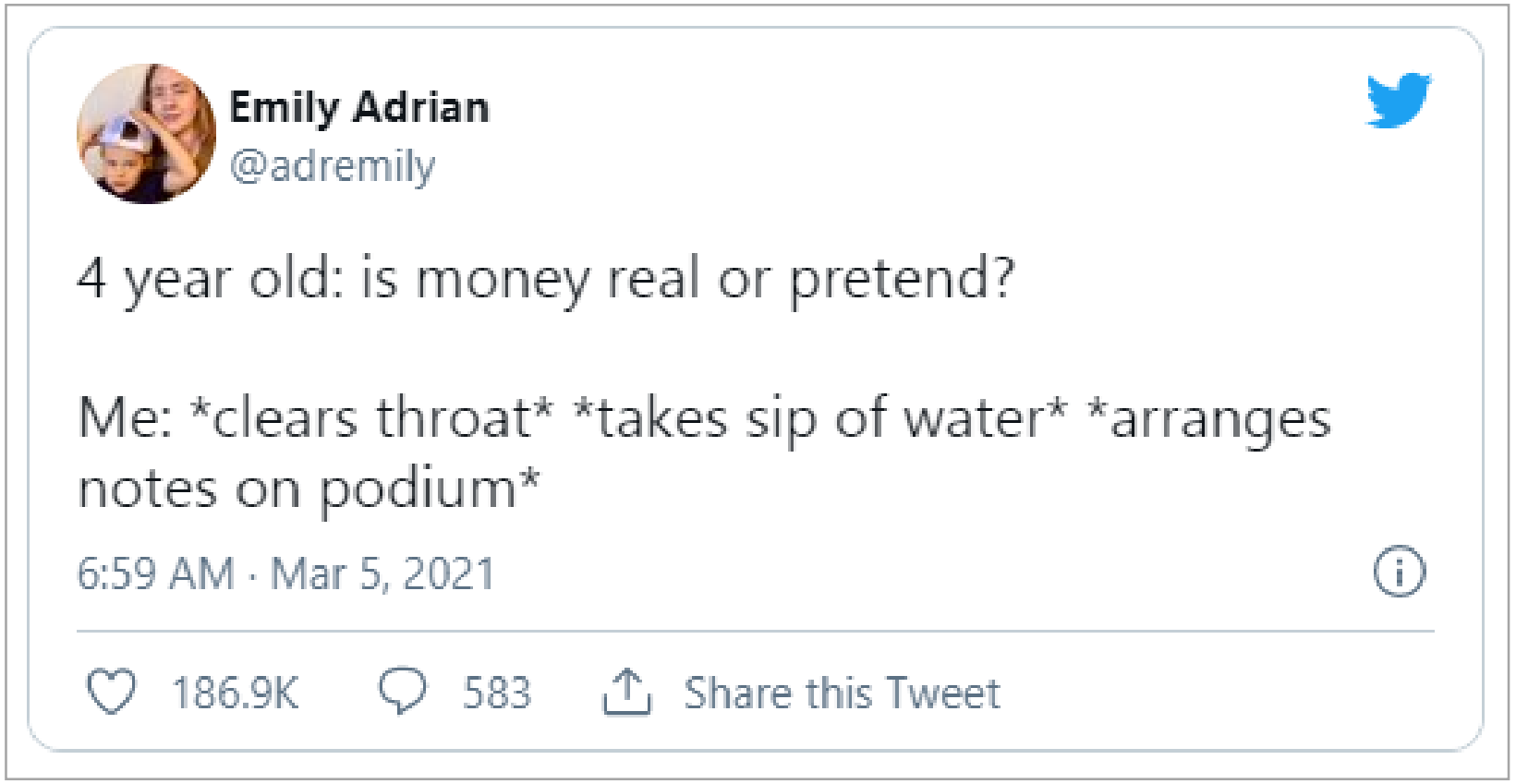
Skin


TOTAL TASKS COMPLETED
Tasks
L
Methods
Things your crewmate can do!
Fuel engines
Download Data
Unlock safe
Walk
qs
TANK
BOX

The reason that we want an object is that we want to associate the Data and Methods together.

We want to group them for many reasons, one of which is organization.






This is a tweet object

Name a piece of data that is part of the tweet object.

Documentation

Search the docs

Twitter API

Getting started

Tutorials

Tools and libraries

Migrate

API reference index

The new Twitter API v2

Fundamentals

Actually, the tweet object is formally and publically defined so programmers can use it.

# Data dictionary

Introduction Object model Using fields and expansions Example payloads

Object model contents

Tweet object

User object

Media object

Poll object

Place objects

## Tweet

Tweets are the basic building block of all things Twitter. The Tweet object has a long list of 'root-level' fields, such as `id`, `text`, and `created_at`. Tweet objects are also the 'parent' object to several child objects including `user`, `media`, `poll`, and `place`. Use the field parameter `tweet.fields` when requesting these root-level fields on the Tweet object.

The Tweet object that can be found and expanded in the user resource. Additional Tweets related to the requested Tweet

This is some of the tweet object's code

Name a category of data that is part of the tweet object.


```
"id": "1212092628029698048",
"text": "Today I had a banana for breakfast",
"author_id": "2244994945",
"public_metrics": {
  "retweet_count": 8,
  "reply_count": 2,
  "like_count": 40,
  "quote_count": 1
},
"lang": "en",
"created_at": "2019-12-31T19:26:16.000Z",
"source": "Twitter Web App",
"in_reply_to_user_id": "2244994945",
```

```
public void save(View view){
 try {
   FileOutputStream out = openFileOutput("data.txt", Activity.MODE_PRIVATE);
   for(int i=0; i<a.length; i++) {
      out.write(a[i]);
   }

   out.flush();
   out.close();
 } catch (FileNotFoundException e) {
   e.printStackTrace();
 } catch (IOException e) {
   e.printStackTrace();
 }
}
```

Take a look at the output file code from yesterday.

## Object FileOutputStream

| Data | Methods |
| --- | --- |
| File name<br>Mode | ? |

Name a file method.

```
public void save(View view){
 try {
   FileOutputStream out = openFileOutput("data.txt", Activity.MODE_PRIVATE);
   for(int i=0; i<a.length; i++) {
      out.write(a[i]);
   }

   out.flush();
   out.close();
 } catch (FileNotFoundException e) {
   e.printStackTrace();
 } catch (IOException e) {
   e.printStackTrace();
 }
}
```

Take a look at the output file code from yesterday.

## Object FileOutputStream

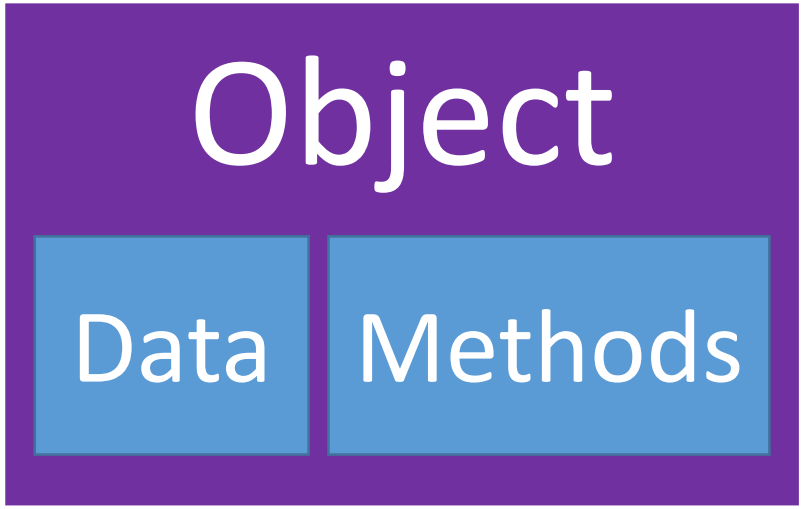
| Data | Methods |
| --- | --- |
| File name | write |
| Mode | flush |
| | close |

Name a file method.

```
<ImageView
    android:layout_width="wrap_content"
    android:id="@+id/flagpic"
    android:src="@drawable/green"
    android:layout_margin="10dp"
    android:layout_gravity="center"
    android:layout_height="wrap_content" />
```

An ImageView is an Object. Name one piece of data it stores. Look in the XML.

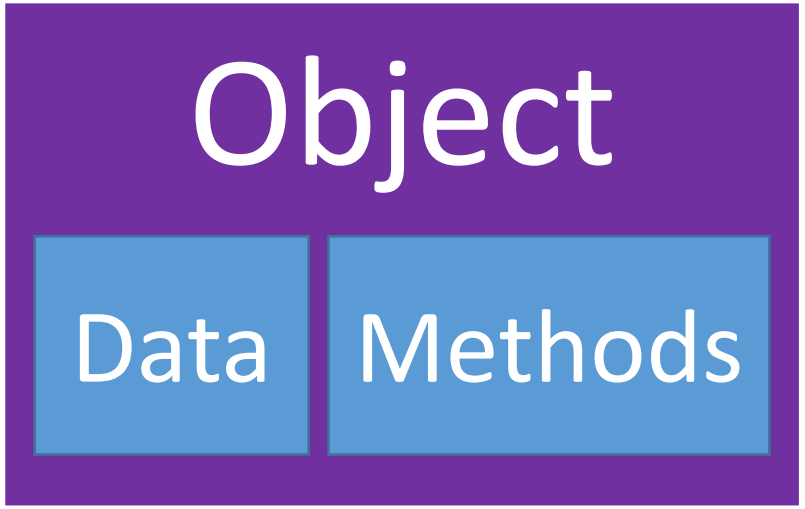
```
pics[m] = new ImageView( context: this);
setpicStart(pics[m], m);
pics[m].setImageResource(R.drawable.green);
pics[m].setId(m);
pics[m].setOnClickListener((v) → {
        gridButtonClick(v.getId());
});
```



```
<ImageView
    android:layout_width="wrap_content"
    android:id="@+id/flagpic"
    android:src="@drawable/green"
    android:layout_margin="10dp"
    android:layout_gravity="center"
    android:layout_height="wrap_content" />
```

An ImageView is an Object. Name one of its methods. Look in the Java.

```
pics[m] = new ImageView( context: this);
setpicStart(pics[m], m);
pics[m].setImageResource(R.drawable.green);
pics[m].setId(m);
pics[m].setOnClickListener((v) → {
        gridButtonClick(v.getId());
});
```



Another way of thinking of an object (that I find useful) is that an object is a complex variable type.

Primitive Types

int, double,
boolean, char

Objects
(Complex Types)

String, ImageView, Button,
FileOutputStream

Where would you classify a TextView?

Primitive Types

Objects (Complex Types)

Where would you classify a Jlabel?

Primitive Types

Objects (Complex Types)

Where would you classify a boolean?

Primitive Types

Objects (Complex Types)

Some differences:

| | Primitive Types | Objects (Complex Types) |
|---|---|---|
| Examples | int, double, boolean, char | String, ImageView, Button, FileOutputStream |

Some differences:

| | Named with small letter<br>Primitive Types | Named with capital letter<br>Objects (Complex Types) |
|---|---|---|
| Examples | int, double, boolean, char | String, ImageView, Button, FileOutputStream |

Some differences:

| | Named with small letter<br>Primitive Types | Named with capital letter<br>Objects (Complex Types) |
|---|---|---|
| Examples | int, double, boolean, char | String, ImageView, Button, FileOutputStream |
| Testing equality | score1 == score2 | score1.equals(score2) |

Some differences:

| | Named with small letter<br>Primitive Types | Named with capital letter<br>Objects (Complex Types) |
|---|---|---|
| Examples | int, double, boolean, char | String, ImageView, Button, FileOutputStream |
| Testing equality | score1 == score2 | score1.equals(score2) |
| Changing value | score1 = 4;<br>score2++; | score1.setText("score:"+ s); |

Some differences:

| | Named with small letter<br>Primitive Types | Named with capital letter<br>Objects (Complex Types) |
|---|---|---|
| Examples | int, double, boolean, char | String, ImageView, Button, FileOutputStream |
| Testing equality | score1 == score2 | score1.equals(score2) |
| Changing value | score1 = 4;<br>score2++; | score1.setText("score:"+ s); |
| Finding value | score1<br>(Use variable name) | score1.getText(); |

The user of our code changes when we start writing Objects.

Our fellow programmers are the users of our objects.

# Object Method Types

Constructor
(new)
sets up memory

Mutator
(set)
changes memory

Accessor
(get, is)
accesses memory

Facilitator
(everything else)
complex tasks

Categorize the method “new TextView”

# Object Method Types

Constructor
(new)
sets up memory

Mutator
(set)
changes memory

Accessor
(get, is)
accesses memory

Facilitator
(everything else)
complex tasks

Categorize the method “compareTo”

# Object Method Types

Constructor
(new)
sets up memory

Mutator
(set)
changes memory

Accessor
(get, is)
accesses memory

Facilitator
(everything else)
complex tasks

Categorize the method “getText”

# Object Method Types

Constructor
(new)
sets up memory

Mutator
(set)
changes memory

Accessor
(get, is)
accesses memory

Facilitator
(everything else)
complex tasks

Categorize the method “equals”

# Object Method Types

Constructor
(new)
sets up memory

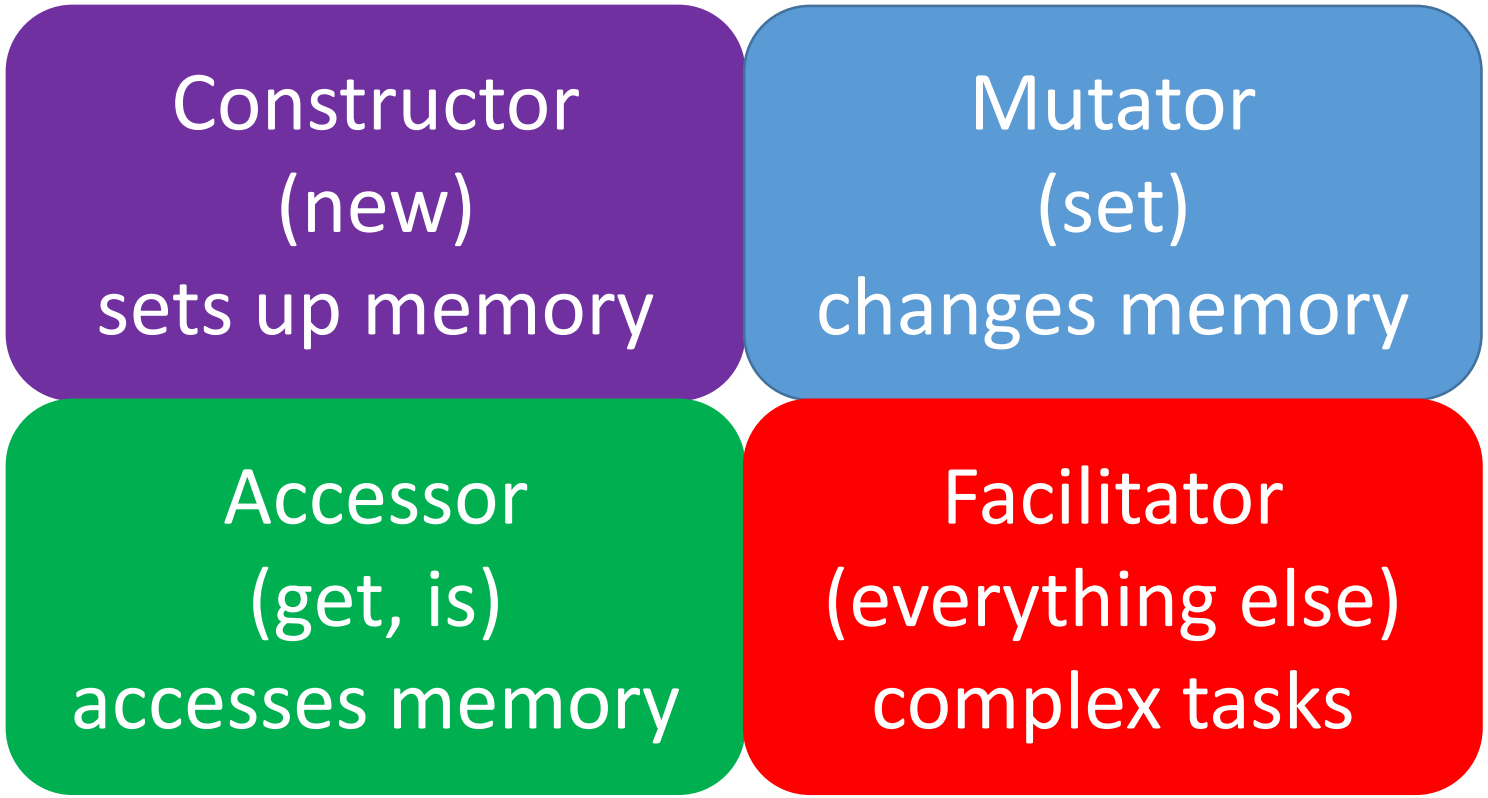
Mutator
(set)
changes memory

Accessor
(get, is)
accesses memory

Facilitator
(everything else)
complex tasks

Categorize the method “setText”

# Object Method Types




Categorize the method "toUpperCase"

# Object Method Types

Constructor
(new)
sets up memory

Mutator
(set)
changes memory

Accessor
(get, is)
accesses memory

Facilitator
(everything else)
complex tasks

Categorize the method “charAt”

# Object Method Types

Constructor
(new)
sets up memory

Mutator
(set)
changes memory

Accessor
(get, is)
accesses memory

Facilitator
(everything else)
complex tasks

Categorize the method “isFull”