Class’ constructor must only receive valid inputs, and, calling a method must not result in invalid state

Avoid instantiating classes with invalid inputs.

```
// this object must not be passed around.
// if you need to allow this behavior
// write a another class.
Email address = new Email(“meh”)
```

Down the road, you will always have to check if the object is valid or not, and, maybe you can write invalid data somewhere.

Also, after calling some method on a instance, if it results in a invalid state, throw an exception and destroy it.

Class’ constructor must only receive valid inputs

You must be able to construct any object at any time by giving valid arguments.

In this case, we are hiding that the resource that the class uses is a File, not a string...so we can’t instantiate this class if we have the file in scope.

```
class Stuff {
  private File file;

  public Stuff(String filepath) {
    this.file = File.open(filepath);
  }
}
```

Class’ constructor must only receive valid inputs

You can always validate or load resources before instantiating a class. Prefer static methods so you can give an appropriate name...

```
class Stuff {
  private File file;

  static public fromFilePath(String path) {
    return new Stuff(File.open(path));
  }

  public Stuff(File file) {
    this.file = file;
  }
}
```