Below are few advantages of static typing:

- Static typing helps in earlier detection of programming mistakes as it allows variable types to be checked at compile time itself. With dynamic type checking, majority of type checking is performed at runtime. Due to this there are chances for programming mistakes to make their way into runtime environment. Let's consider the following example in Python, which is dynamically typed. Here x is first initialized with numeric 10, then let's say after few statements due to some carelessness x is **re-initialized** with a string "java". Now, x is a string. Finally, after few more statements, we have the last statement where programmer is computing the value for y by doing 12 – x, i.e., programmer is clearly expecting x to be a numeric value. But, here as x in reality is a string, this statement y = 12 – x; will give an error when program executes. In contrast with Java, we would have declared x as int x = 10;. But, the (re-assignment) statement x = "java", would have given a compilation error as Java does not allow this as type is declared as int and so it cannot be re-assigned a String literal. This is happening at compile time and not at runtime and so static type checking is helping in earlier detection of the programming mistake. This way programmer can fix that mistake right away instead of having to discover it at runtime as in the case of Python.

  ```
  x = 10;

  …

  x = "java";

  …

  y = 12 - x;
  ```

- Static typing is also helpful during method overloading. With method overloading, we can have multiple versions of the same method in the same class. For example, the following two methods are named as ***foo*** in the same class. This is possible because the methods accept different input data, i.e., one is accepting an integer as input while the other is accepting character

data. Without static type checking, method overloading wouldn't be possible. Method overloading will be discussed later in the chapter while discussing methods.

```
void foo(int i) { … }

void foo(char c) { … }
```

- Static typing also permits better developer experience in IDEs such as Eclipse. For example, once we type a class name followed by a dot operator, eclipse will automatically display a ***drop-down menu*** displaying the variables and methods that the class has. This way we can simply choose the particular variable or method without having to type-in the method or variable name. We will see this when we switch to Eclipse.

- Maintainability is another advantage. Some people believe that ***code refactoring*** is tedious in dynamically-typed languages like JavaScript. Note that code refactoring means ***restructuring*** existing code without changing its external behavior

Static vs Dynamic Typed Languages

Checkout the links in references section, which are from the Stackoverflow Website. They compare static vs dynamic typed languages and explain what they mean. Pros & cons of both types of languages are also debated.

References

[1] http://stackoverflow.com/questions/125367/dynamic-type-languages-versus-static-type-languages

[2] http://stackoverflow.com/questions/42934/what-do-people-find-so-appealing-about-dynamic-languages

[3] http://stackoverflow.com/questions/1517582/what-is-the-difference-between-statically-typed-and-dynamically-typed-languages