TASK

# Interview Preparation: Interfaces

Visit our website

## Introduction

### WELCOME TO THE INTERVIEW PREPARATION: INTERFACES TASK!

The most important step in landing your dream job is acing your interview. Interviews allow you to impress your future employer by demonstrating your knowledge and skills. However, an interview can be a nerve-wracking process. Taking the time to prepare for your interview can ensure that the process runs smoothly and is less stressful. In this task, we will be discussing the common interview topic, interfaces.

Get in touch

**Connect for support**

Remember that with our courses, you're not alone! You can contact an expert code reviewer to get support on any aspect of your course.

The best way to get help is to login to Discord at **https://discord.com/invite/hyperdev** where our specialist team is ready to support you.

Our team is happy to offer you support that is tailored to your individual career or education needs. Do not hesitate to ask a question or for additional support!

In this task, two programming languages that use very different approaches to software development will be compared: Python, which is an object-oriented programming language, and JavaScript, a prototype-based language. Both languages are commonly used for web development. Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Python is a strongly-typed language whereas JavaScript is a loosely-typed language. How these differences affect your code and the pros and cons of each difference are briefly introduced below. Once these concepts are clear, it will be easier to understand what interfaces are and when they are useful.

## DIFFERENCES IN TYPING

Both JavaScript and Python are dynamically typed languages as opposed to statically typed languages. With ***statically typed*** languages, you have to specify what data type a variable will contain when you declare it. If you declare a variable as a number in a statically typed language and later try to assign that variable a value that is a letter, it would throw an error. However, with ***dynamically typed*** languages, object types are determined at runtime. Therefore, you can declare a variable that can hold a string at one point in time but a number at another stage.

Python is strongly-typed, whereas JavaScript is loosely-typed. With a ***strongly-typed*** language, you are not allowed to do anything that's incompatible with the type of data with which you're working. For example with Python, if you tried to add 3 and “Hello”, this would result in an error because, with a strongly-typed language, you can’t add a string and a number to each other. However, with JavaScript, you could add a string and a number because a ***loosely-typed*** language tries to convert data types to allow you to perform the instruction given. Sometimes, the fact that JavaScript is loosely-typed can lead to unexpected results that can cause errors in code.

**TypeScript** is a typed superset of JavaScript. TypeScript makes JavaScript a statically-typed language instead of a loosely-typed language. TypeScript is designed by Microsoft and compiles into plain JavaScript.

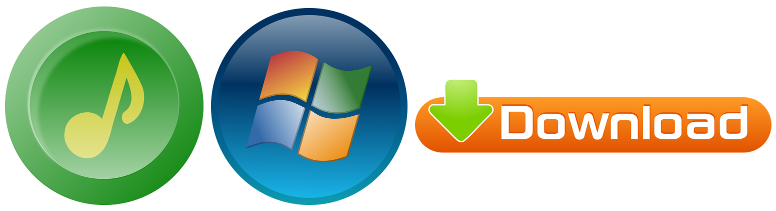
## OOP VS PROTOTYPE BASED

Object-oriented programming (OOP) is an extremely popular and useful programming paradigm. Many programming languages are OOP languages. Therefore, as a software developer, you must understand the basics of OOP.

Here is a VERY quick recap of what OOP is:
An object is created using a class. A class is a blueprint from which objects are made. It consists of both data and the code that manipulates the data.

To illustrate, let's consider an object you encounter every time you use software, a button. As shown in the images below, there are many different kinds of buttons with which you might wish to interact. All these objects have certain things in common:

- They are described by certain ***attributes***, i.e. every button has a size, background colour and shape. It could also have a specific image on it or it could contain text.
- You expect all buttons to have ***methods*** that do something, i.e. when you click on a button, you want something to happen - you may want a file to download or an app to launch. The code that tells your computer what to do when you click on the button is written in a method.



Although every object is different, you can create a single class that describes all objects of a similar kind. The ***class*** defines the attributes and methods that each object created using that class must contain. Each ***object*** is called an *instance of a class*. Each of the buttons shown in the image above is an instance of the class button.

OOP has become so popular because of the benefits associated with this way of coding including code reuse, shorter development times, increased productivity, and easier maintenance and testing of systems.

Now that we have had a quick recap on OOP theory, let's compare Python and JavaScript. Python is an OOP language. As previously mentioned, with Python, we define classes that are used to create objects. JavaScript, on the other hand, is an object-based scripting language. It also works with objects but with JavaScript, objects are created using ***prototypes*** instead of classes. A prototype is basically a constructor function. Also, remember that since ES2015 changes to JavaScript, we can create classes. However, ES2015 classes are a simple sugar over the prototype-based OO pattern. In other words, ES2015 allows us to create objects using keywords (like class, super, etc.) that are used in class-based languages but under the hood, ES2015 still uses functions and prototypal inheritance to implement objects.

| Python | JavaScript |
|---|---|
| • Strongly-typed language<br>• Object-oriented language<br>  ○ Objects created using classes | • Loosely-typed language<br>• Object-based language<br>  ○ Objects created using prototypes (constructor functions) |

## INTRODUCTION TO INTERFACES

With the theory that we have just covered in mind, let's turn our attention to interfaces. What are interfaces? Interfaces are a key concept in OOP. A class can implement an interface. An *interface defines which methods and properties a class must implement*. The interface itself does not contain any code to implement an object. An interface will, for example, specify that a class that is built based on it should contain a method called "doSomething()" but won't contain any code to implement it. Interfaces specify what a class must do but not how they go about doing it. An interface is often referred to as a contract because it is agreed that a class that implements the interface will implement the properties and methods defined in the interface.

The code below shows an example of an interface in Python. As you can see, the interface defines two methods that any class that implements the interface must implement. The interface does not implement the methods. It just shows that these methods must be implemented in the class.

```
from interface import Interface

class MyInterface(Interface):

    def method1(self):
        pass

    def method2(self, arg1, arg2):
        pass
```

An example of a class that implements the interface is shown below:

```
from interface import implements

class MyClass(implements(MyInterface)):

    def method1(self):
        return "method1"

    def method2(self, arg1, arg2):
        return "method2"
```

*The 2 code examples above are from* ***https://pypi.python.org/pypi/python-interface/1.2.0***

Interfaces are used to achieve abstraction. ***Abstraction*** involves showing only the relevant data and hiding all unnecessary details of an object from the user. The methods in an interface do not contain bodies. They have to be implemented by a class before you can access them. Therefore, the interface hides all unnecessary implementation from the user.

Another key reason that software developers use interfaces is to ensure that all classes are built according to rules, or contracts, outlining what methods and properties must be implemented. Using interfaces is also a way of enforcing ***strict typing*** to a certain degree.

Python supports the implementation of interfaces because it is an OOP language. On the other hand, *JavaScript has no built-in way of creating or implementing interfaces*. Since interfaces are a contract that defines what classes should look like and JavaScript doesn't use classes to create objects this makes sense. However, JavaScript is very flexible and there are several ways in which we can achieve the goal of implementing interfaces using JavaScript. For example, TypeScript can also be used to create JavaScript that uses classes and interfaces.

## BEFORE YOU START YOUR COMPULSORY TASK

In this task, you are again going to be asked to create a blog. Remember that this task aims to prepare you to find a job in the real world. Since these interview preparation tasks are meant to help convey your knowledge of a topic to a prospective employer, a blog is a good format to help you get used to explaining technical concepts to others. Keep your blog post conversational, as if you were explaining the topic to someone during a job interview.

Also remember that as a web developer, you are going to need to be able to continue learning and researching. To help develop these skills, you are asked to do some research in these interview preparation tasks. These tasks enable you to hone your research skills and be fully prepared as you get ready to start working as a web developer.

## Compulsory Task 1

Follow these steps:

- **Create a blog post called "Interfaces in object oriented programming languages and prototype-based languages" that includes the following:**
    - Explain what an interface is and the benefits of using interfaces in OOP.
    - Explain why JavaScript does not really use interfaces and how objects are created with JavaScript. Use JavaScript code snippets in your explanation. **Recommended reading**.
    - Explain how you could emulate interfaces using JavaScript. **Recommended reading**.
    - Explain what strict mode is in JavaScript and why you would use this. **Recommended reading**.
    - Explain what TypeScript is and how it can be used to create interfaces (**Recommended reading**) and enforce strict typing in JavaScript (**Recommended reading**).

If you are having any difficulties, please feel free to contact our specialist team **on Discord** for support.

## Completed the task(s)?

Ask an expert to review your work!

**Review work**

Rate us

## Share your thoughts

HyperionDev strives to provide internationally-excellent course content that helps you achieve your learning outcomes.

Think that the content of this task, or this course as a whole, can be improved, or think we’ve done a good job?

**Click here** to share your thoughts anonymously.