**Compiling Your First Smart Contract**

**Compiler directive:**

The pragma directive specifies the *version* of the Solidity compiler that you want to use to build your source file. When the compiler encounters this line, it will check its version against the one you specified here. If the compiler version is different, Remix will automatically adjust accordingly to your specifications.

**You can specify the compiler version(s) in the following ways:**

1. use exactly one version

pragma solidity 0.8.19; // use only version 0.8.19

1. use versions that fall within a lower and upper range

// use versions between 0.8.19 and 0.9.0 (excluded)

pragma solidity ^0.8.19;

pragma solidity >=0.8.19 <0.9.0;

**SPDX License Identifier:**

It's a good practice (even not mandatory) to start your smart contract with an SPDX License Identifier. It helps in making licensing and sharing code easier from a legal perspective.

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.19;

The MIT license is recognized as one of the most permissive, granting anyone the freedom to use the following code and essentially use it as they see fit.

**Writing the Smart Contract:**

You can start writing your contract using the keyword contract followed by a name, e.g., SimpleStorage. All the code inside the curly brackets will be considered part of this contract.

If you are familiar with Object Oriented Programming languages, you can think of a *contract* as a concept similar to a *class*.

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.19;

contract SimpleStorage {

//this is where we'll place the content of the contract

}

**Compiling:**

1. In Remix IDE, select the Solidity Compiler.
2. Choose the version of the compiler that matches the version specified in your Solidity file.
3. Hit the Compile button.

Compiling your code means taking human-readable code and transforming it into computer-readable code or bytecode.

If you see a green checkmark, it means your compilation was successful. If there is any error, Remix will point out where the error is, and you can debug it accordingly.