Value Types in Solidity:

Introduction
In Solidity, value types are basic data types that are passed by value. This means that when a variable of a value type is assigned to another variable, a copy of the value is made. This is different from reference types, which are passed by reference.

Key Value Types

1. **Boolean (`bool`)**
   - Represents a binary value (`true` or `false`).
   - Example:
     ```solidity
     bool isActive = true;
     ```

2. **Integers**
   - **Signed Integers (`int`):** Can hold both positive and negative values.
     - Sizes: `int8`, `int16`, `int24`, ..., `int256` (in steps of 8 bits).
     - Example:
       ```solidity
       int8 smallNumber = -10;
       int256 bigNumber = 123456789;
       ```
   - **Unsigned Integers (`uint`):** Can hold only non-negative values.
     - Sizes: `uint8`, `uint16`, `uint24`, ..., `uint256` (in steps of 8 bits).
     - Example:
       ```solidity
       uint8 smallPositiveNumber = 10;
       uint256 largePositiveNumber = 123456789;
       ```

3. **Fixed Point Numbers**
   - Currently not fully supported in Solidity.
   - Placeholder for future use: `fixed`, `ufixed`.

4. **Address**
   - Holds 20-byte Ethereum addresses.
   - Example:
     ```solidity
     address wallet = 0x1234567890123456789012345678901234567890;
     ```

5. **Bytes**
   - **Fixed-size Byte Arrays:** Ranging from `bytes1` to `bytes32`.
     - Example:
       ```solidity
       bytes32 data = "SolidityBytes32";
       ```
   - **Dynamic-size Byte Arrays (`bytes`):** More flexible, can change size.

    - Example:
      ```solidity
      bytes dynamicData = "SolidityBytes";
      ```

6. **Enum**
   - User-defined type to represent a set of named values.
   - Example:
     ```solidity
     enum State { Pending, Active, Inactive }
     State currentState = State.Pending;
     ```

7. **Function Type**
   - Represents a function within the current contract.
   - Example:
     ```solidity
     function myFunction(uint x) public pure returns (uint) {
        return x * 2;
     }
     function () external myFunc = myFunction;
     ```

Properties and Usage
- **Default Values:**
  - `bool` defaults to `false`.
  - `int` and `uint` default to `0`.
  - `address` defaults to `0x0000000000000000000000000000000000000000`.
  - `bytes` defaults to an empty byte array.
  - `enum` defaults to the first value (index `0`).
- **Operators:**
  - Arithmetic: `+`, `-`, `*`, `/`, `%`
  - Comparison: `==`, `!=`, `<`, `>`, `<=`, `>=`
  - Logical: `&&`, `||`, `!`
  - Bitwise (for integers and bytes): `&`, `|`, `^`, `~`, `<<`, `>>`

Examples

Boolean
```solidity
bool isReady = false;
isReady = !isReady; // isReady is now true
```

Integers
```solidity
uint8 u = 255; // max value for uint8
int256 i = -100;
```

Address
```solidity
address owner = 0x1234567890123456789012345678901234567890;
```

Bytes
```solidity
bytes32 fixedData = "FixedData";
bytes dynamicData = "DynamicData";
```

Enum
```solidity
enum Status { New, Approved, Rejected }
Status currentStatus = Status.New;
```

Conclusion
Understanding value types in Solidity is fundamental for writing efficient and error-free smart contracts. These types form the building blocks for more complex data structures and are essential for managing state and behavior in contracts.