**Solidity Math**

**Introduction:**

In this lesson, we will guide you through converting the value of ETH to USD. We’ll use the previously defined `getPrice` function within the new `getConversionRate` function.

**The `getPrice` and `getConversionRate` Functions:**

The `getPrice` function returns the current value of Ethereum in USD as a `uint256`.

The `getConversionRate` function converts a specified amount of ETH to its USD equivalent.

**Decimal Places:**

In Solidity, only integer values are used, as the language does not support floating-point numbers.

Function getConversionRate(uint256 ethAmount) internal view returns (uint256) {

Uint256 ethPrice = getPrice();

Uint256 ethAmountInUsd = (ethPrice \* ethAmount) / 1e18;

Return ethAmountInUsd;

}

**Note:**  The line `uint256 ethAmountInUsd = (ethPrice \* ethAmount)` results in a value with a precision of 1e18 \* 1e18 = 1e36. To bring the precision of `ethAmountInUsd` back to 1e18, we need to divide the result by 1e18.

**Caution:** Always multiply before dividing to maintain precision and avoid truncation errors. For instance, in floating-point arithmetic, `(5/3) \* 2` equals approximately 3.33. In Solidity, `(5/3)` equals 1, which when multiplied by 2 yields 2. If you multiply first `(5\*2)` and then divide by 3, you achieve better precision.

**Example of `getConversionRate`:**

* `ethAmount` is set at 1 ETH, with 1e18 precision.
* `ethPrice` is set at 2000 USD, with 1e18 precision, resulting in 2000e18.
* `ethPrice \* ethAmount` results in 2000e36.
* To scale down `ethAmountInUsd` to 1e18 precision, divide `ethPrice \* ethAmount` by 1e18.

**Checking Minimum USD Value:**

We can verify if users send at least 5 USD to our contract:

Require(getConversionRate(msg.value) >= MINIMUM\_USD, “You need to spend more ETH!”);

Since `getConversionRate` returns a value with 18 decimal places, we need to multiply `5` by `1e18`, resulting in `5 \* 1e18` (equivalent to `5 \* 10\*\*18`).

**Deployment to the Testnet:**

In Remix, we can deploy the `FundMe` contract to a testnet. After deployment, the `getPrice` function can be called to obtain the current value of Ethereum. It’s also possible to send money to this contract, and an error will be triggered if the ETH amount is less than 5 USD.

// Gas estimation failed. Error execution reverted, didn’t send enough ETH.

**Conclusion:**

In this lesson, we’ve demonstrated how to convert ETH to USD using the `getConversionRate` function, ensuring precise calculations by handling decimal places correctly.

**Updates:**

Math Clarification: 1:12

Last updated on September 23, 2024

Whoops! We meant to say 36 decimal places!