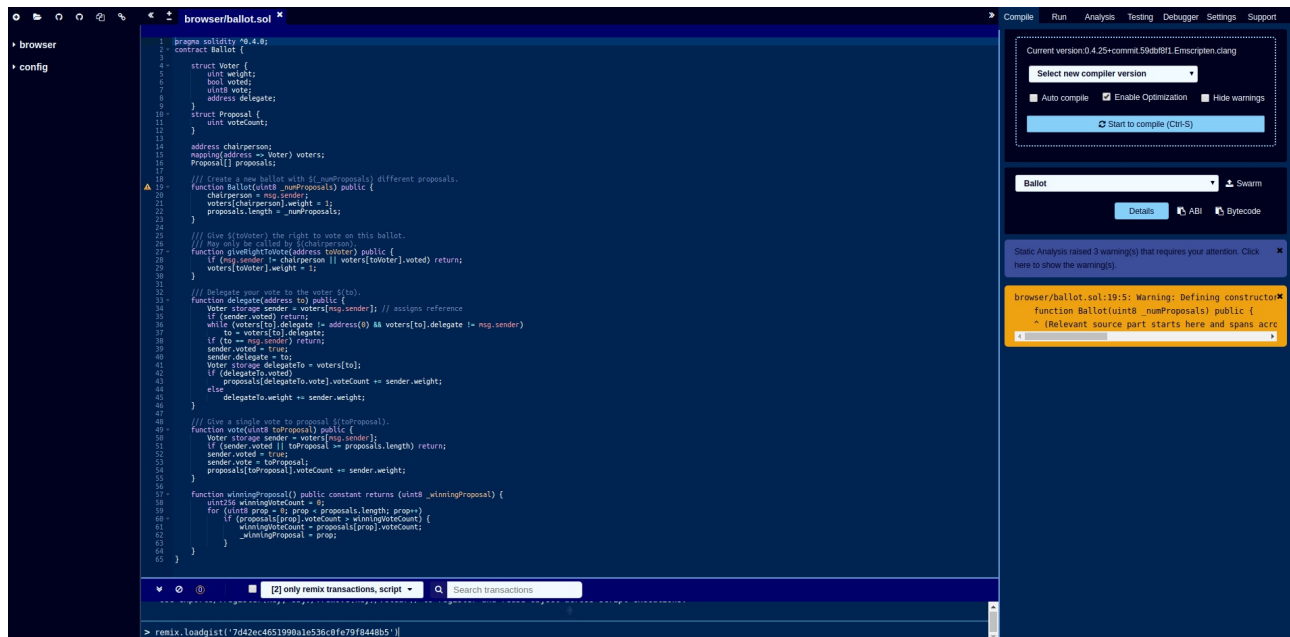


Presentation (how to script contract deployment and contract interaction)

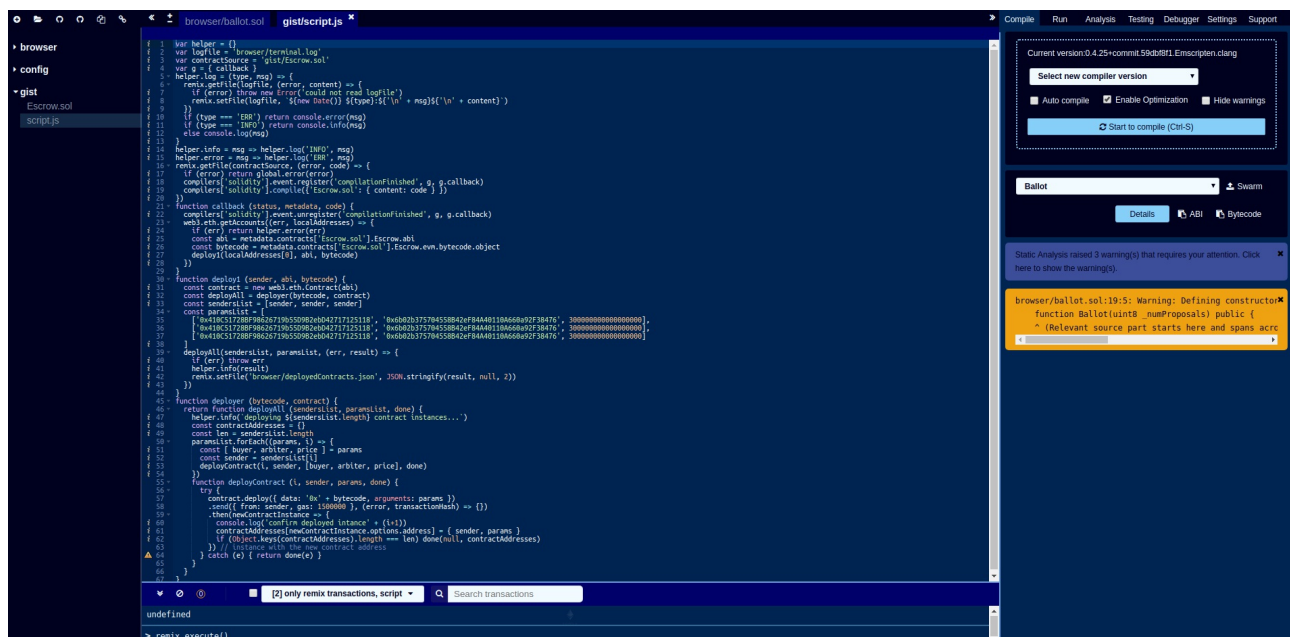
Deploy many contract instances of the same contract with different constructor inputs

- gist: 7d42ec4651990a1e536c0fe79f8448b5

1. load gist with files from terminal



2. execute script



3. see deployedContracts.json to see result

The image shows the Remix IDE interface. At the top, there are tabs for 'browser', 'baloot.sol', 'baloot_test.sol', 'deployedContracts.json', 'terminal.log', 'config', 'Escrow.sol', and 'script.js'. The left sidebar contains a tree view with 'browser', 'baloot.sol', 'baloot_test.sol', 'terminal.log', 'config', 'Escrow.sol', and 'script.js'. The main editor area displays a Solidity contract named 'Escrow' with the following code:

```
contract Escrow {
    address payable owner;
    address payable recipient;
    uint256 amount;
    uint256 deadline;
    bool released;

    constructor(address payable _owner, address payable _recipient, uint256 _amount, uint256 _deadline) public {
        owner = _owner;
        recipient = _recipient;
        amount = _amount;
        deadline = _deadline;
        released = false;
    }

    function confirm() public {
        require(owner == msg.sender, "Only owner can confirm");
        released = true;
    }

    function confirm_deployed() public {
        require(owner == msg.sender, "Only owner can confirm");
        released = true;
    }

    function confirm_deployed3() public {
        require(owner == msg.sender, "Only owner can confirm");
        released = true;
    }
}
```

The right sidebar shows the 'Current version 0.4.25-commit.5' and 'Emscripten clang' compiler settings. The bottom panel shows the deployment progress, including the transaction hash and the 'remix' logo.