**发送ETH方法**

**接收ETH合约**

我们先部署一个接收ETH合约ReceiveETH。ReceiveETH合约里有一个事件Log，记录收到的ETH数量和gas剩余。还有两个函数，一个是receive()函数，收到ETH被触发，并发送Log事件，另一个是查询合约ETH余额的getBalance()函数。

|  |
| --- |
| Solidity contract Receive {  //收到eth事件，记录amount和gas  event Log(uint amount,uint gas);   //receive方法，接收eth时被触发  receive() external payable{  emit Log(msg.value, gasleft());  }   //返回合约ETH余额  function getBalance() view public returns(uint) {  return address(this).balance;  } } |

**发送ETH合约**

首先，现在发送ETH合约SendETH中实现payable的构造函数和receive()，让我们能够在部署时和部署后向合约转账。

|  |
| --- |
| Solidity contract SendETH {  //构造函数，payable使得部署的时候可以转eth进去  constructor() payable{}  //receive方法，接收eth时被触发  receive() external payable{} |

**transfer方法**

1.用法是接收方地址.transfer(发送ETH数额)。

2.transfer()的gas限制是2300，足够用于转账，但对方合约的fallback()或receive()函数不能实现太复杂的逻辑

3.transfer()如果转账失败，会自动revert(回滚交易)

|  |
| --- |
| Solidity //用transfer()发送ETH，\_to填Receive合约的地址，amount是ETH转账金额 function transferETH(address payable \_to, uint256 amount) external payable{  \_to.transfer(amount); } |

**send方法**

1.用法是接收方地址.send(发送ETH数额)。

2.send()的gas限制是2300，足够用于转账，但对方合约的fallback()或receive()函数不能实现太复杂的逻辑。

3.send()如果转账失败，不会revert。

4.send()的返回值是bool，代表着转账成功或失败，需要额外代码处理一下。

|  |
| --- |
| Solidity function sendETH(address payable \_to, uint256 amount) external payable{  //处理下send的返回值，如果失败，revert交易并发送error  bool success = \_to.send(amount);  if(!success){  revert SendFailed();  } } |

**call方法**

1.用法是接收方地址.call{value: 发送ETH数额}("")

2.call()没有gas限制，可以支持对方合约fallback()或receive()函数实现复杂逻辑

3.call()如果转账失败，不会revert

4.call()的返回值是(bool， data)，其中bool代表转账成功或失败，需要额外代码处理

|  |
| --- |
| Solidity function callETH(address payable \_to, uint256 amount) external payable{  //处理下call的返回值，如果失败，revert交易并发送error  (bool success,) = \_to.call{value:amount}("")  if(!success){  revert CallFailed()；  } } |

**总结**

三种发送ETH的方法：transfer，send和call。

* call没有gas限制，最为灵活，是最提倡的方法；
* transfer有2300 gas限制，但是发送失败会自动revert交易，是次优选择；
* send有2300 gas限制，而发送失败不会自动revert交易，几乎没有人用它。