# Lab: Basics of Contracts

## Contract ownership

Write a contract, that:

* knows its owner (contract publisher)
* has a method that **changes the owner**
  + can **only be called by the current owner**
  + the method takes one argument - **address** of the new owner
  + at each owner change, a contract **event is created** with two fields - **the old** and **new owner**
* has a method that **accepts the ownership**
  + can only be executed by the address that is set from the method that changes the owner
  + the owner has only 30 seconds to accept the ownership
* the fallback function shall emit an event with the **sender** and the **value** send with the transaction (amount of Ethers)

## Auction

The Bulgarian government wants to sell the Bulgarian Railways and they want to use Ethereum Smart Contract for the auction. Your job is to provide a smart contract that can handle the auction without any financial fraud and “shurobadjanashtina”.

Create an auction contract, that:

* Has an owner.
* Has method to place bid
  + The auction should not be cancelled or expired
  + Should accept ETH
  + Everyone except the owner can place new bid.
  + The bid should be greater than the highest bid otherwise the function should throw an exception
* Has method to check the highest bid and the highest bidder.
  + use the correct function modifier
* Has method to cancel the auction.
  + Only the owner can cancel the auction
* Has method to withdraw funds
  + Should send ETH.
  + If the auction is cancelled everyone can withdraw their bid.
  + If the auction is not cancelled but it’s expired, then the owner of the auction can withdraw the highest bidder bid. Everyone else can withdraw their bid.
* Every method should emit an event.
* Use assert and require whenever possible.
* The constructor shall receive 2 parameters – start time and end time.
  + You need to validate them correctly. Think about it.
* Create custom modifiers which will help you with all functions.