

Errors ▾

[Error Triage](#)[Error Guide](#)[Exercise](#)

The new Keyword ▾

[Creating New Contracts](#)[Step by Step Guide](#)[Exercise](#)[Contract to Contract
Interactions](#) >[Events](#) >[Address and Payable](#) >**Development with Foundry**[Deploying a smart contract
using Foundry](#)[Foundry: Setting up Foundry](#)[The new Keyword](#)

The New Keyword

[Copy page](#) ▾

Learn to create a contract that creates other contracts.

You've seen the `new` keyword and used it to instantiate `memory` arrays with a size based on a variable. You can also use it to write a contract that [creates other contracts](#).

Objectives

By the end of this lesson you should be able to:

Write a contract that creates a new contract with the new keyword

Creating a Simple Contract Factory

A contract factory is a contract that creates other contracts. To start, let's create and interact with a very simple one. Create a new project in Remix and add a file called `ContractFactory.sol`.

Adding the Template

Imagine you want to create a contract that can store its owner's name and complement it upon request. You can create this contract fairly easily.

Reveal code

Deploy and test.

Creating a Factory

The `Complimenter` contract is a huge success! People love how it makes them feel and you've got customers banging on the doors and windows. Awesome!

The only problem is that it takes time and effort to manually deploy a new version of the contract for each customer. Luckily, there's a way to write a contract that will act as a self-service portal for your customers.

Error Triage

Error Guide

Exercise

Creating New Contracts

Step by Step Guide

Exercise

Development with Foundry

Deploying a smart contract
using Foundry

Foundry: Setting up Foundry

Start by adding a contract called `ComplimenterFactory` . The Remix interface makes things easier if you leave the factory in the same file as `Complimenter` .

Add a function called `CreateComplimenter` that is public, accepts a `string` called `_name` , and returns an `address` .

Creating a new contract is simple: `new Complimenter(_name)`

You can also save the return from that instantiation into a variable. This reference can be used to call public functions in the deployed contract, and can be cast to an address. We can use it to get an easy reference to find the copies made by the factory. The end result should look similar to:

Reveal code

Testing

Clear the environment if you haven't already, then start by deploying `ComplimenterFactory` . You've been working hard and deserve nice things, so call `CreateComplimenter` with your name.

In the terminal, the *decoded output* will be the address of the new contract.



Error Triage

Error Guide

Exercise



Creating New Contracts

Step by Step Guide

Exercise

Development with Foundry

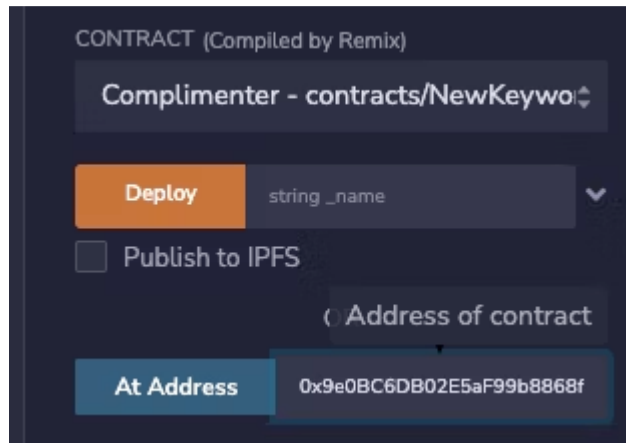
Deploying a smart contract
using Foundry

Foundry: Setting up Foundry

```
{  
  "0": "address: 0x9e0BC6DB02E5aF99b8868f0b732eb45c956B92dD"  
}
```

Copy **only** the address.

Switch the *CONTRACT* to be deployed to **Complimenter** , then paste the address you copied in the field next to the *At Address* button which is below the *Deploy* button.



Click *At Address* and the instance of **Complimenter** should appear below **ComplimenterFactory** . Test to confirm it works, then try deploying more instances with the factory.

Error Triage

Error Guide

Exercise

Creating New Contracts

Step by Step Guide

Exercise

Development with Foundry

Deploying a smart contract
using Foundry

Foundry: Setting up Foundry

Deployed Contracts



Q Search...

Ctrl K

GitHub

Support

Base Build



Get Started

Base Chain

Base Account

Base App

Mini Apps

OnchainKit

Cookbook

Showcase

Learn

Error Triage

Error Guide

Exercise



Creating New Contracts

Step by Step Guide

Exercise

Development with Foundry

Deploying a smart contract
using Foundry

Foundry: Setting up Foundry



Low level interactions



CALLDATA

Transact



COMPLIMENTER AT 0X9E0...B92DD



Balance: 0 ETH

compliment

0: string: You look great today, Brian

name



If the deployed contract appears, but is instead a broken copy of the factory, it's because you didn't change the contract in the *CONTRACT* dropdown above the deploy button.

Remix is trying to interact with `Complimenter` using the *ABI* from the factory contract, which won't work.

Ask a question

Ctrl+Q

Conclusion

In this lesson, you learned how to deploy contracts from another contract by using the `new` keyword. You also learned that you look great today!

Was this page helpful?



Yes



No



Suggest edits



Raise issue

[< Creating New Contracts](#)

[Exercise >](#)



[base.org](#)

[Blog](#)

[Privacy Policy](#)

[Terms of Service](#)

[Cookie Policy](#)

Development with Foundry

Deploying a smart contract
using Foundry

Foundry: Setting up Foundry