Building Starknet Smart Contracts

In the previous section, we gave an introductory example of a smart contract written in Cairo, describing the basic blocks to build smart contracts on Starknet. In this section, we'll be taking a deeper look at all the components of a smart contract, step by step. When we discussed [_interfaces_][contract interface], we specified the difference between the two types of _public functions_, i.e., _external functions_ and _view functions_, and we mentioned how to interact with the _storage_ of a contract.

At this point, you should have multiple questions that come to mind:

- How can I store more complex data types?
- How do I define internal/private functions?
- How can I emit events? How can I index them?
- Is there a way to reduce the boilerplate?

Luckily, we'll be answering all these questions in this chapter. Let's consider the `NameRegistry` contract in Listing {{#ref reference-contract}} that we'll be using throughout this chapter:

```cairo,noplayground
{{#include ../listings/ch14-building-starknet-smart-contracts/
listing_01_reference_contract/src/lib.cairo:all}}
```

{{#label reference-contract}}
Listing {{#ref reference-contract}}: Our reference contract for this chapter

[contract interface]: ./ch13-02-anatomy-of-a-simple-contract.md#the-interface-the-contracts-blueprint