Appendix E - Common Error Messages

You might encounter error messages when writing Cairo code. Some of them occur very frequently, which is why we will be listing the most common error messages in this appendix to help you resolve common issues.

- `Variable not dropped.`: this error message means that you are trying to make a variable with a type that do not implement the `Drop` trait go out of scope, withtout destroying it. Make sure that variables that need to be dropped at the end of the execution of a function implement the `Drop` trait or the `Destruct` trait. See [Ownership](ch04-01-what-is-ownership.md#destroying-values---example-with-feltdict) section.
- `Variable was previously moved.`: this error message means that you are trying to use a variable whose ownership has already been transferred to another function. When a variable doesn't implement the `Copy` trait, it is passed by value to functions, and ownership of the variable is transferred to the function. Such a variable cannot be used anymore in the current context after its ownership has been transferred. It is often useful to use the `clone` method in this situation.
- `error: Trait has no implementation in context: core::fmt::Display::<package_name::struct_name>`: this error message is encountered if you try to print an instance of a custom data type with `{}` placeholders in a `print!` or `println!` macro. To mitigate this issue, you need to either manually implement the `Display` trait for your type, or use the `Debug` trait by applying `derive(Debug)` to your type, allowing to print your instance by adding `:?` in `{}` placeholders.
- `Got an exception while executing a hint: Hint Error: Failed to deserialize param #x.`: this error means that the execution failed because an entrypoint was called without the expected arguments. Make sure that the arguments you provide when calling an entrypoint are correct. There is a classic issue with `u256` variables, which are actually structs of 2 `u128`. Therefore, when calling a function that takes a `u256` as argument, you need to pass 2 values.
- `Item path::item is not visible in this context.`: this error message lets us know that the path to bring an item into scope is correct, but there is a vibility issue. In cairo, all items are private to parent modules by default. To resolve this issue, make sure that all the modules on the path to items and items themselves are declared with `pub(crate)` or `pub` to have access to them.
- `Identifier not found.`: this error message is a bit aspecific but might indicate that:
  - A variable is being used before it has been declared. Make sure to declare variables with the `let` keyword.
  - The path to bring an item into scope is wrongly defined. Make sure to use valid paths.

Starknet Components Related Error Messages

You might encounter some errors when trying to implement components. Unfortunately, some of them lack meaningful error messages to help debug. This section aims to provide you with some pointers to help you debug your code.

- `Trait not found. Not a trait.`: this error can occur when you're not importing the component's impl block
  correctly in your contract. Make sure to respect the following syntax:

  ```cairo,noplayground
  #[abi(embed_v0)]
  impl IMPL_NAME = PATH_TO_COMPONENT::EMBEDDED_NAME<ContractState>
  ```

```

- `Plugin diagnostic: name is not a substorage member in the contract's Storage. Consider adding to Storage: (...)`: the compiler helps you a lot debugging this by giving you recommendation on the action to take. Basically, you forgot to add the component's storage to your contract's storage. Make sure to add the path to the component's storage annotated with the `#[substorage(v0)]` attribute to your contract's storage.
- `Plugin diagnostic: name is not a nested event in the contract's Event enum. Consider adding to the Event enum:` similar to the previous error, the compiler tells you that you forgot to add the component's events to your contract's events. Make sure to add the path to the component's events to your contract's events.
```