Part 20 - Structure of a Boo Project

Part 20 - Structure of a Boo Project

On the Project-level

Here I'll use the example of the IRC bot I write: Goomba

```
+ Goomba (Goomba namespace)
   |+ Configuration (Goomba.Configuration
namespace)
   |   |- Config.boo
   |       |# class Config
   |+ Data (Goomba.Data namespace)
   |   |- Column.boo
   |   |   |# class Column
   |   |- Database.boo
   |   |   |# enum DatabaseType
   |   |   |# class Database
   |   |- DatabasePreferences.boo
   |   |   |# class DatabasePreferences
   |   |- Result.boo
   |       |# class Result
   |+ Plugins (Goomba.Plugins namespace)
   |   |- DefineCommand.boo
   |   |   |# class DefineCommand
   |   |       |# class Definition
   |   |- Hail.boo
   |   |   |# class Hail
   |   |       |# class HailMessage
   |   |- HelpCommand.boo
   |   |   |# class HelpCommand
   |   |- Logger.boo
```

```
    |   |   |# class Logger
    |   |       |# class Message
    |   |       |# class Action
    |   |- Quoter.boo
    |   |   |# class Quoter
    |   |       |# class Quote
    |   |- RawLogger.boo
    |   |   |# class RawLogger
    |   |- UrlGenerator.boo
    |   |   |# class UrlGenerator
    |   |       |# class Engine
    |   |- UserTracker.boo
    |   |   |# class UserTracker
    |   |       |# class User
    |   |- VersionCommand.boo
    |   |   |# class VersionCommand
    |   |- UrlTracker.boo
    |       |# class UrlTracker
    |           |# class Url
    |- ActionEventArgs.boo
    |   |# enum ActionType
    |   |# class ActionEventArgs
    |- DebugLogger.boo
    |   |# enum LogImportance
    |   |# class DebugLogger
    |- Goomba.boo
    |   |# class Goomba
    |   |! Main Body (This will be executed
when Goomba.exe is run)
    |- GoombaPreferences.boo
    |   |# class GoombaPreferences
```

```
|- IPlugin.boo
|   |# interface IPlugin
|- MessageEventArgs.boo
|   |# enum MessageType
|   |# class MessageEventArgs
|- Sender.boo
```

```
|# enum SenderType
|# class Sender
```

Which I have set up to create the assemblies `Goomba.exe`, `Goomba.Data.dll`, `Goomba.Configuration.dll`, as well as one assembly per plugin.

You may have noticed a few important things:

- For every directory, it represents a different namespace, with the same name as the directory itself.
- Each `.boo` file has at most one class in it. That class will have the **same exact** name as the `.boo` file.
- The "Main Body" section is below the `class Goomba` definition. Any inline executable code must be at the bottom of a file in the assembly.
- `Enums` come before `classes`. This is merely a coding practice that is not required, but recommended. If an `enum` is larger than 15 values, place it in its own file.

On the File-level

Files must be defined in this order:

1. Module docstring
2. Namespace declaration
3. Import statements
4. `Enums/Classes/Structs/Interfaces`
5. Functions
6. Main code executed when script is run
7. Assembly attributes

> ✅ **Recommendation**
>
> One class per file. If you have more than one class per file, split it up.
> If you have a class inside another class, this is acceptable, as it still has one flat class per file.

Go on to **Part 21 - Documentation**