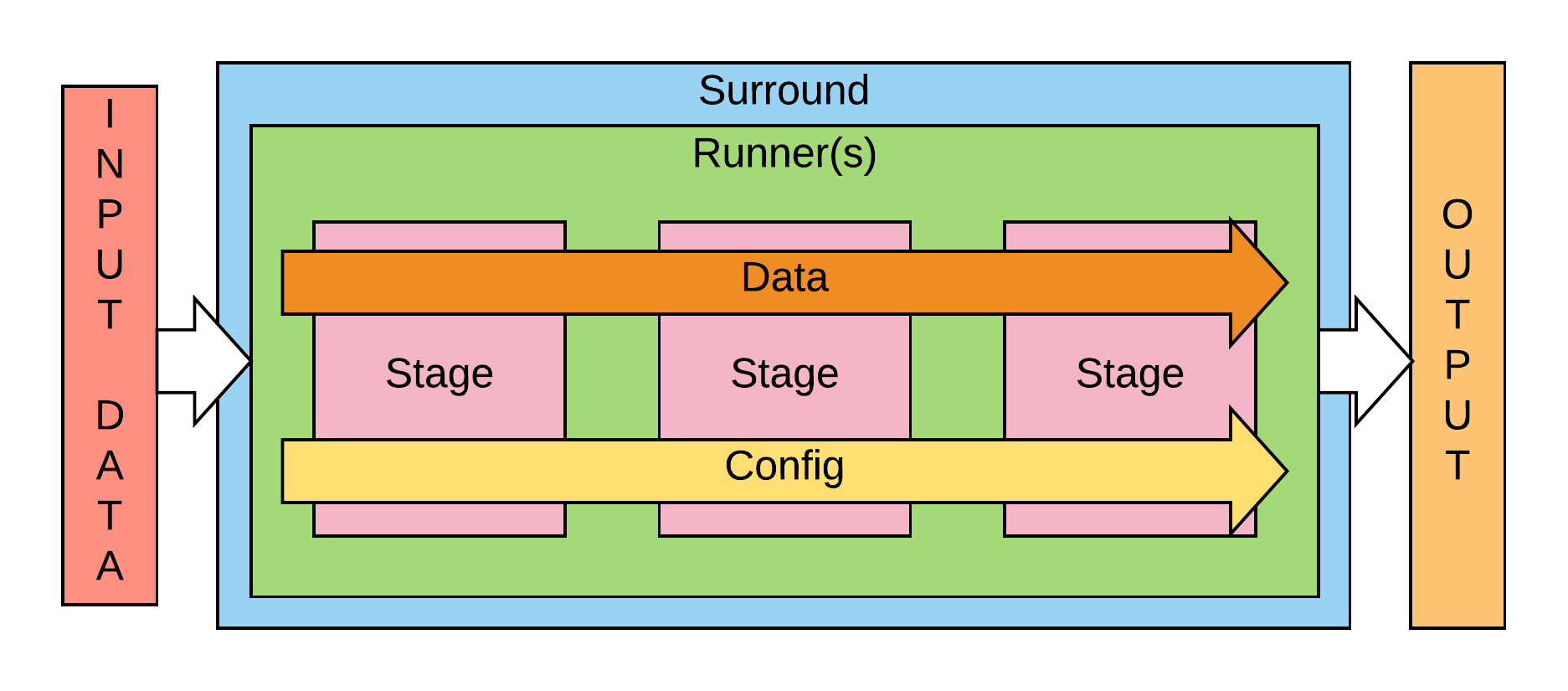
Data Science with Surround

**Surround** is introduced as a lightweight framework. It serves as a pipeline for machine learning in Python. What Surround is offering is flexibility and simplicity for the data scientists. Which aims to help them focus more on the problems and less on glue code.

**How it works**

****

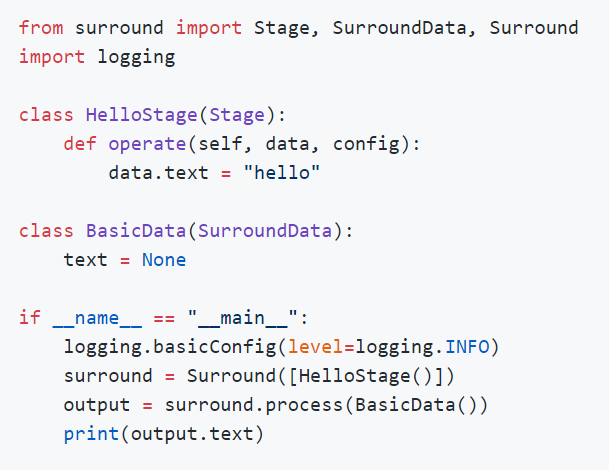
*Surround* can be a combination of stages or just one depending on the need of the user. The stages can be arranged in any order the user sees fit and these stages process and transform the input data into meaningful output data. The *surround data* is shareable between and contains information for each stage. What each stage does it read and process the surround data and put back the processed output for the next stage to use that data. This goes one till the final stage which gives us the meaningful output data.

While working with surround data the thing to keep in mind is that a user can create as many variable as he want but the four core variables that comes with surround data are stage\_metadata, execution\_time, errors and warnings.

*Surround stage* are used for data transformation. It’s the surround stages where the surround data are processed and modified. Each stage can be set to perform a number of actions which helps it process the input data to an extend that the output data of that stage can be used by the next stage. This goes on until the last stage which gives the user desired output.

The number of pipeline implementation is also in the hand of the user which helps him interchange between pipelines.

**An example for reference**



This example shows how to print the text "hello" to the screen using surround. The process consists of defining the operation in the *operate()* method of *HelloStage*. The object being processed is an instance of *BasicData*, which inherits from *SurroundData*. The surround object is initialised with only one stage as surround = *Surround([HelloStage()]*, and the line output = *surround.process(BasicData())* calls the *operate()* method of *HelloStage*, using a new instance of *BasicData* as a parameter. Finally, the content of the *BasicData* object is printed to screen.

**Using Surround**

We need to understand the goals that surround is trying to achieve in order to understand under which circumstances does one need surround to use. The problems that surround is aiming to overcome are

* The same changes were required again and again to refactor code written by data scientists to make it ready for serving e.g. no standard way to run scripts, no standard way to handle configuration and no standard pipeline architecture.
* Existing model serving solutions focus on serving the model rather than serving an end-to-end solution. Our machine learning projects require multiple models and glue code to tie these models together.
* Existing serving approaches do not allow for the evolution of a machine learning pipeline without re-engineering the solution i.e. using a cloud API for the first release before training a custom model much later on.
* Code was commonly being commented out to run other branches as experimentation was not a first class citizen in the code being written.

And under any of the following circumstances one should consider using surround

* You want a flexible way to serve a pipeline in Python without writing C/C++ code.
* You have multiple models (custom or pre-trained) from different frameworks that need to be combined into a single Surround solution.
* You want to use existing intelligent APIs (AWS Rekognition, Google Cloud AI, Cognitive Services) as part of your Surround implementation.
* You have pre or post processing steps that aren't part of your models but need to be deployed as part of your Surround implementation.
* You need to package up your dependencies for running Surround as an offline solution on another machine.

In other words the goal of surround is simply to make the life of data scientists easier and at the same time give them the opportunity to work on their tasks without worrying about the codes they are leaving behind.