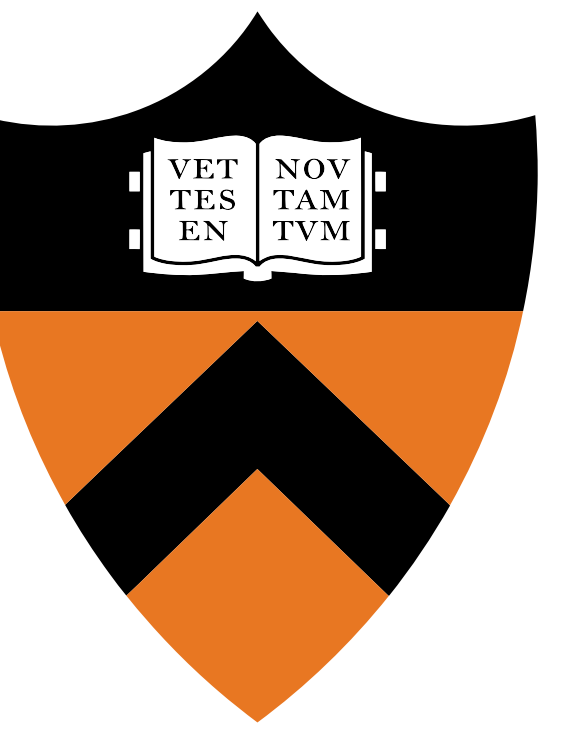




Describe Data to get Science-Data-Ready Tooling: Awkward Target for KSY

Manasvi Goyal¹, Ianna Osborne¹, Jim Pivarski¹, Amy Roberts² Andrea Zonca³

1. Princeton University, Princeton, NJ, USA; 2. University of Colorado Denver, Denver, CO, USA; 3. San Diego Supercomputer Center, La Jolla, CA, USA



Need and Motivation

Scientific data formats can differ across experiments due to specialized hardware and data acquisition systems. The increase in custom data formats has posed a major challenge for collaborations like CDMS that spend hours writing their own tools to read and analyze their data. This project provides a simple solution. Collaborations only need to describe their custom data formats in KSY just once and then directly convert their data into Awkward Arrays using `kaitai_struct_awkward_runtime` API.

What is Kaitai Struct YAML (KSY)?

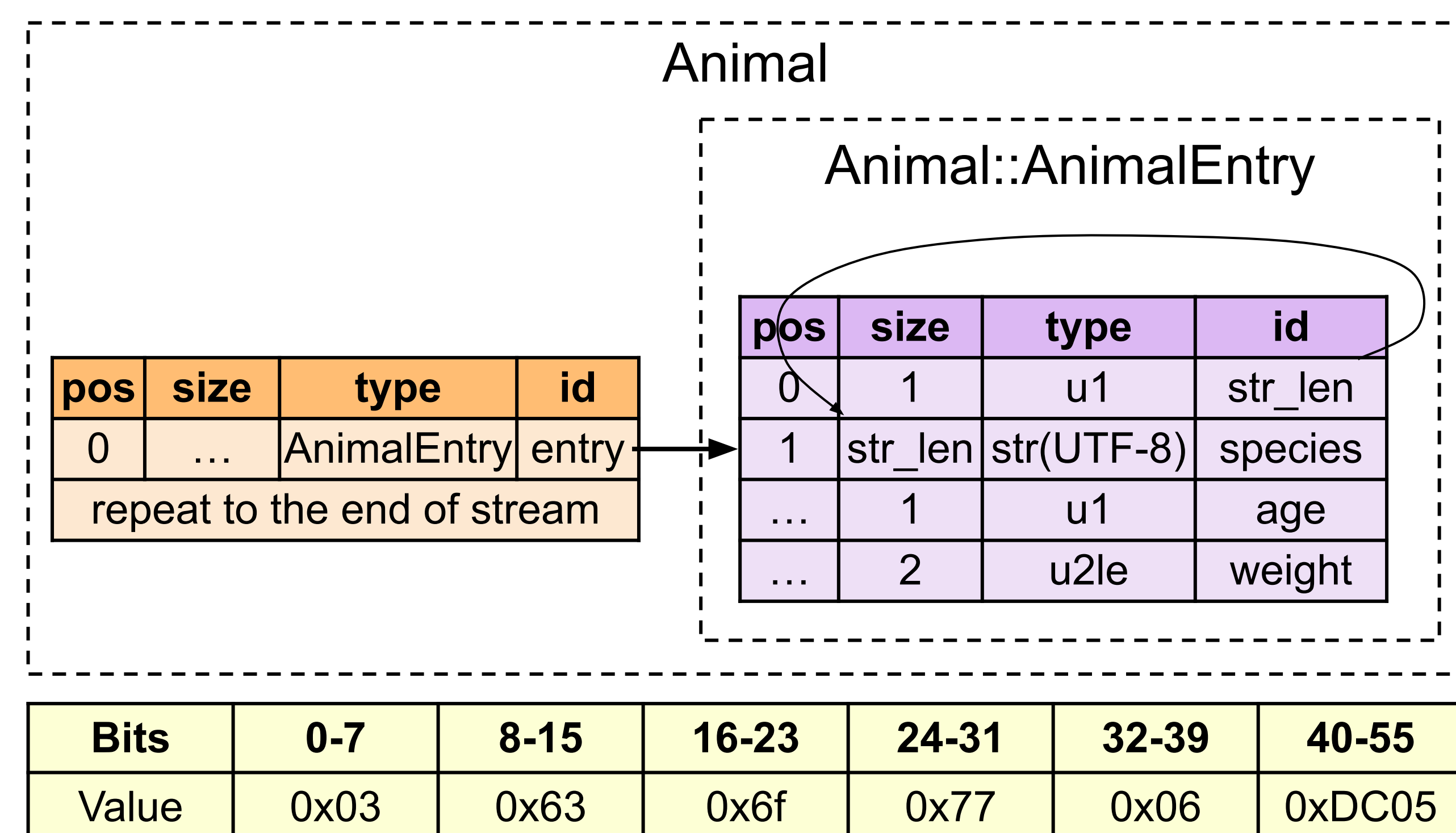
KSY is a declarative language that takes YAML-like descriptions of a data format structure and generates code in any of the supported languages to read a raw data file.



- Compile KSY with `kaitai_struct_compiler` into source files to read the structure in the languages of your choice.
- Utilize `kaitai_struct_[language]_runtime` API to write your own `main()` function to use these libraries for analysis.

Example: animal.ksy

Here is a simple data structure that describes the animal data. However, the actual formats of scientific data are more complex.



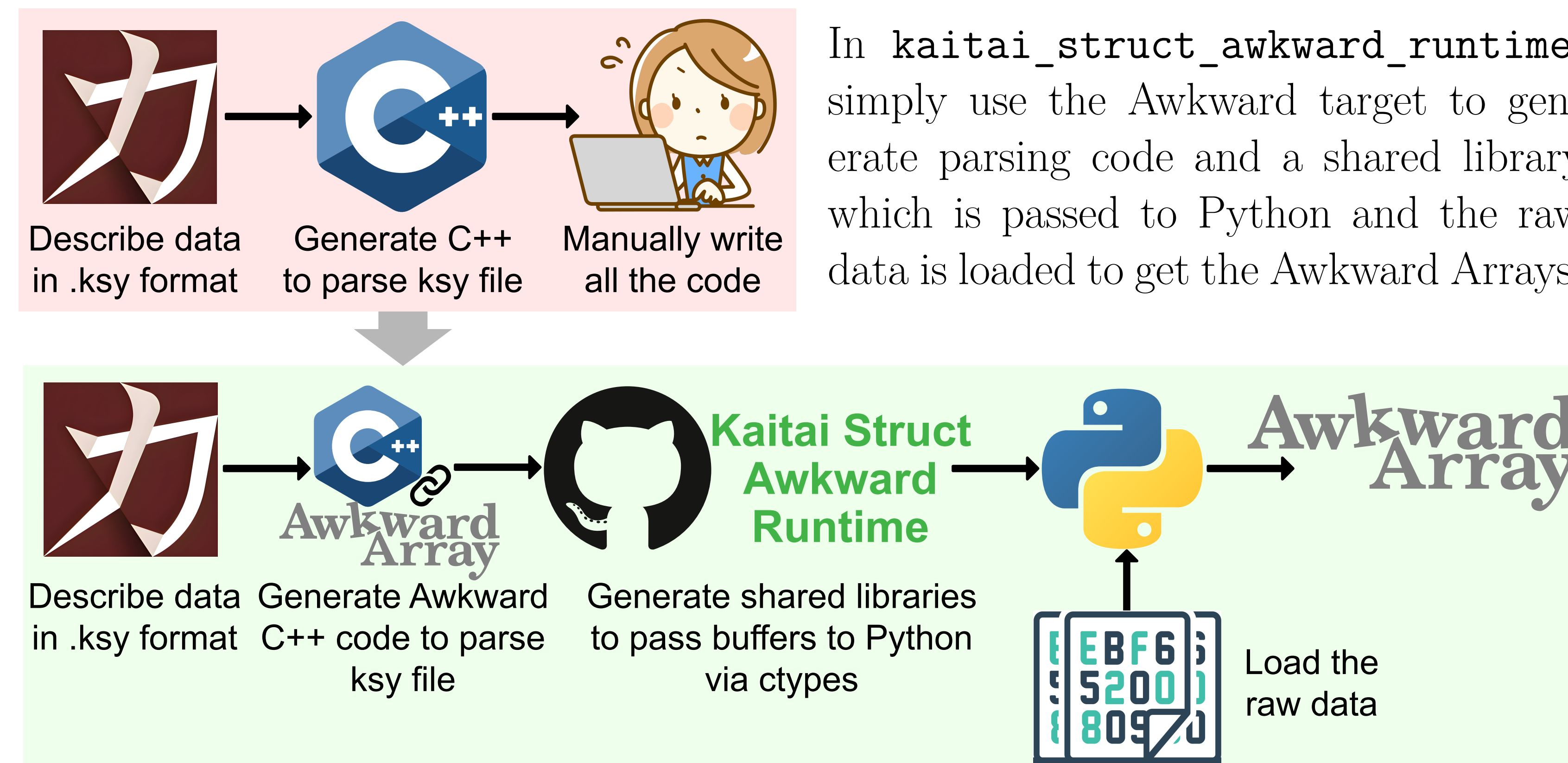
If you don't have a hex chart handy, this entry describes a 6 year old cow that weighs 1500 pounds.

Why Awkward Arrays?

When dealing with large files and complicated data structures, even the most efficient Python code can be quite time and resource-heavy. Awkward Arrays offer a dynamic and efficient approach to represent complex data structures in NumPy-like arrays. Awkward arrays store data in jagged nested arrays of arbitrary types and variable lengths.

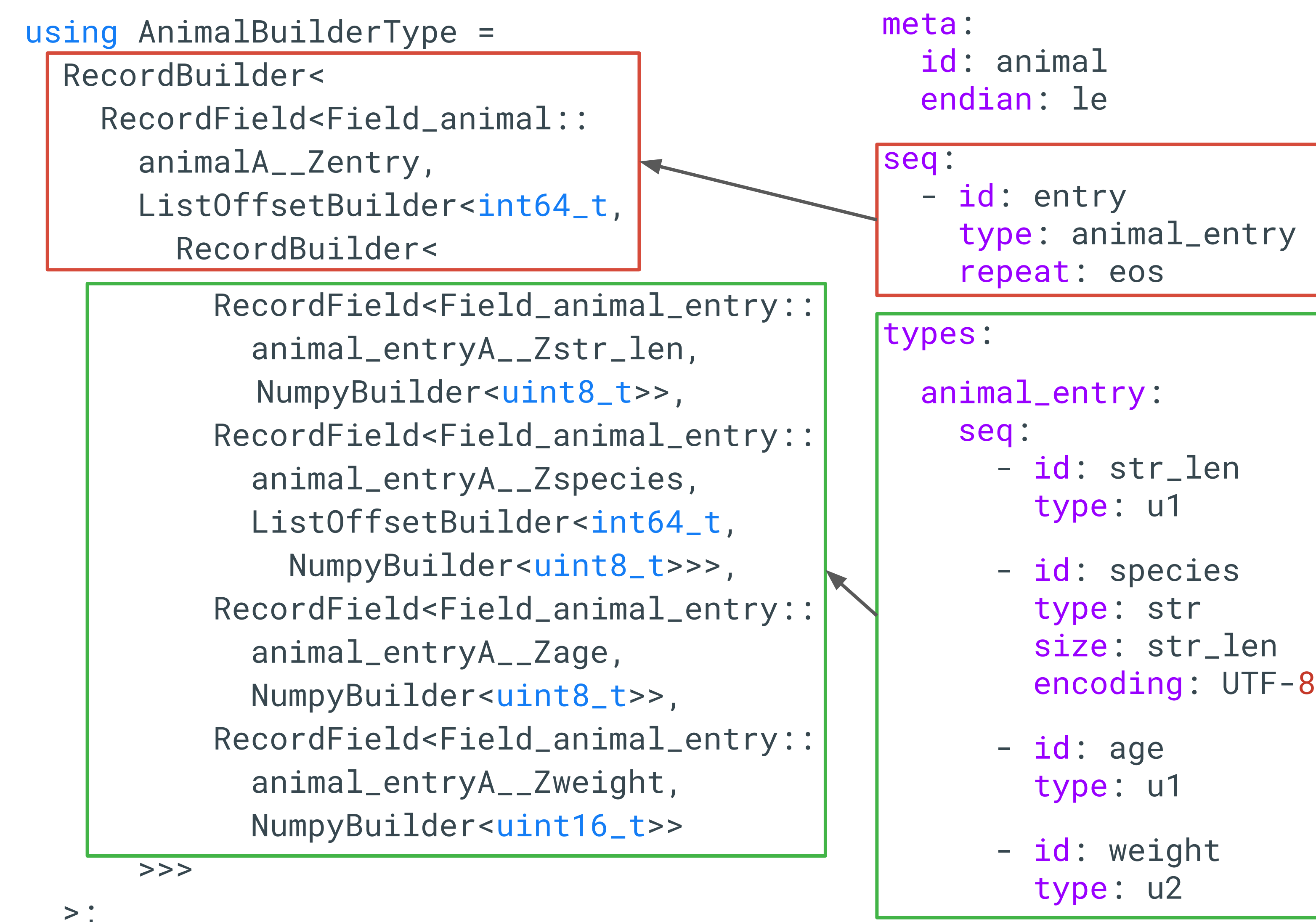
Awkward Target for KSY: User Interface

Describe your custom data format into KSY just once. With just Kaitai, you have to write all the analysis code including `main()`. This takes a lot of time and efforts for complex nested data structures of scientific data.



In `kaitai_struct_awkward_runtime`, simply use the Awkward target to generate parsing code and a shared library which is passed to Python and the raw data is loaded to get the Awkward Arrays.

KSY → LayoutBuilder in animal.ksy



kaitai_struct_awkward_runtime Steps

Clone, install `awkward-kaitai`, generate the C++ files for Awkward target and build `awkward_kaitai` for the main source file.

```

TERMINAL
> git clone --recursive https://github.com/ManasviGoyal/kaitai_struct_awkward_runtime.git
> ./kaitai-struct-compiler -t awkward --outdir src-animal example_data/schemas/animal.ksy
> pip install .
> awkward-kaitai-build src-animal/animal.cpp -b build
  
```

Open Python and print the returned `ak.Array`:

```

import awkward_kaitai
animal = awkward_kaitai.Reader(
    "./src-animal/libanimal.so"
) # pass the shared library
awkward_array = animal.load(
    "example_data/data/animal.raw"
) # pass the raw data file
awkward_array.to_list()[ :2 ]
  
```

Finally, `animal.ksy` is represented in Awkward Arrays as:

```

[ { 'animalA__Zentry': [
  { 'animal_entryA__Zstr_len': 3,
    'animal_entryA__Zspecies': 'cat',
    'animal_entryA__Zage': 5,
    'animal_entryA__Zweight': 12 },
  { 'animal_entryA__Zstr_len': 3,
    'animal_entryA__Zspecies': 'dog',
    'animal_entryA__Zage': 3,
    'animal_entryA__Zweight': 43 }
] } ]
  
```

Acknowledgement

This work is supported by NSF cooperative agreements OAC-1836650 and PHY-2323298 (IRIS-HEP) and grants OAC-2104003 (PONDD) and OAC-2103945 (Awkward Array).