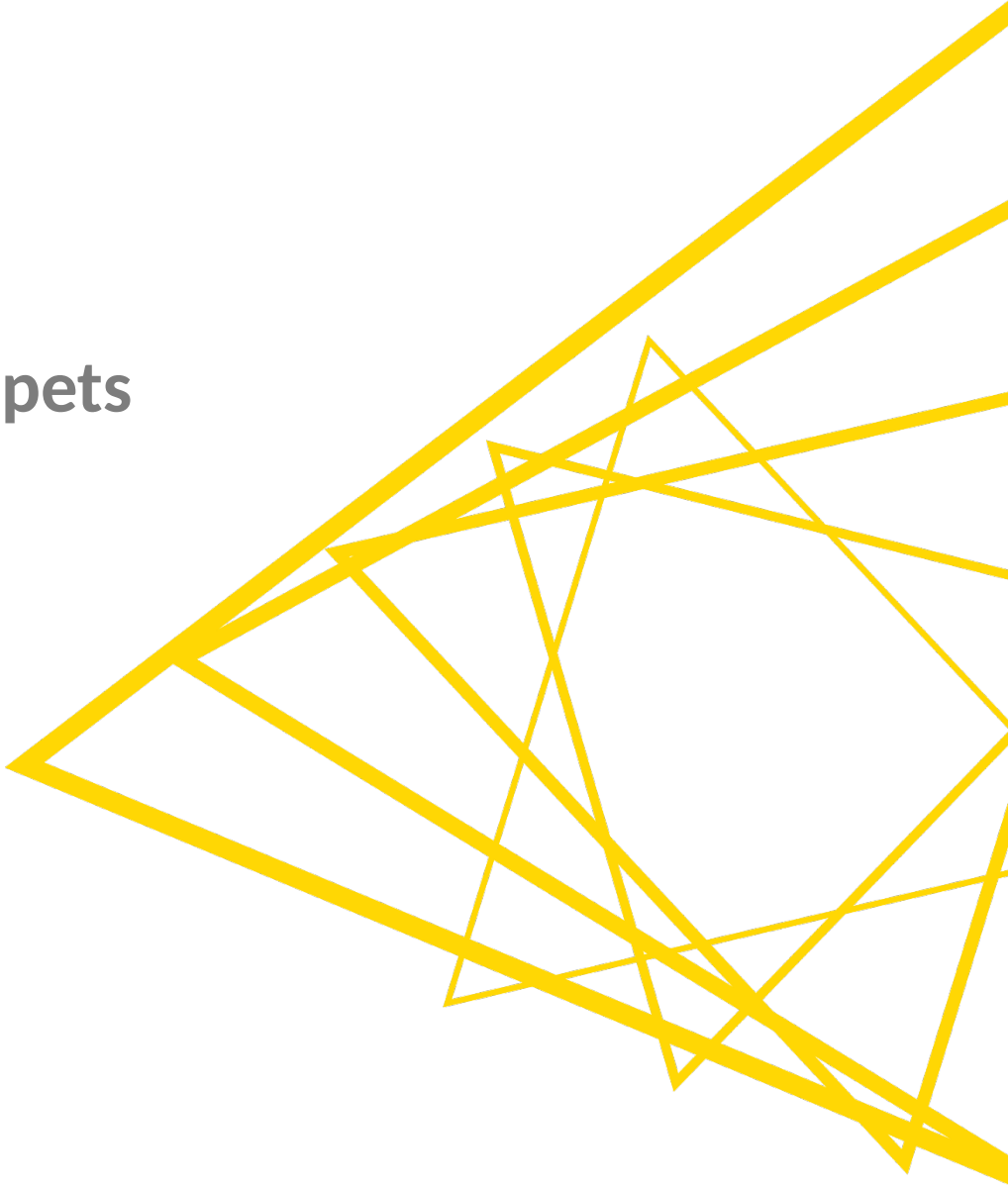
Custom Types in Java Snippets

Jonathan Hale
Intern, KNIME



Custom DataTypes

- KNIME AP has an **Extension Point** to register a custom **DataType**
- Requires implementing `DataCellFactory`

Custom Types in Java Snippets

Currently cannot be used in Java Snippets

Custom Type Support in Java Snippets – Extension Point

One interface to implement for the Extension Point itself:

```
public interface ConverterProvider {

     /**
      * @return a Collection of factories to create “KNIME DataValue -> Java type”
      *         converters
      */
     Collection<ConverterFactory<?, ?>> getConverterFactories();

     /**
      * @return a Collection of factories to create “Java type -> KNIME DataType”
      *         converters
      */
     Collection<ToDataValueConverterFactory<?>> getToDataValueConverterFactories();

}
```

Custom Type Support in Java Snippets – Extension Point

Sample implementation code snippet:

```
@Override
public Collection<ToDataValueConverterFactory> getToDataValueConverterFactories() {
     return Arrays.asList(
          new SimpleToDataValueConverterFactory(
               Integer.class,      // input type
               IntCell.TYPE,  // output type
               (source) -> new IntCell(source) // conversion function
          )
     );
}
```

Custom Type Support in Java Snippets – Annotations

In an existing DataCellFactory implementation:

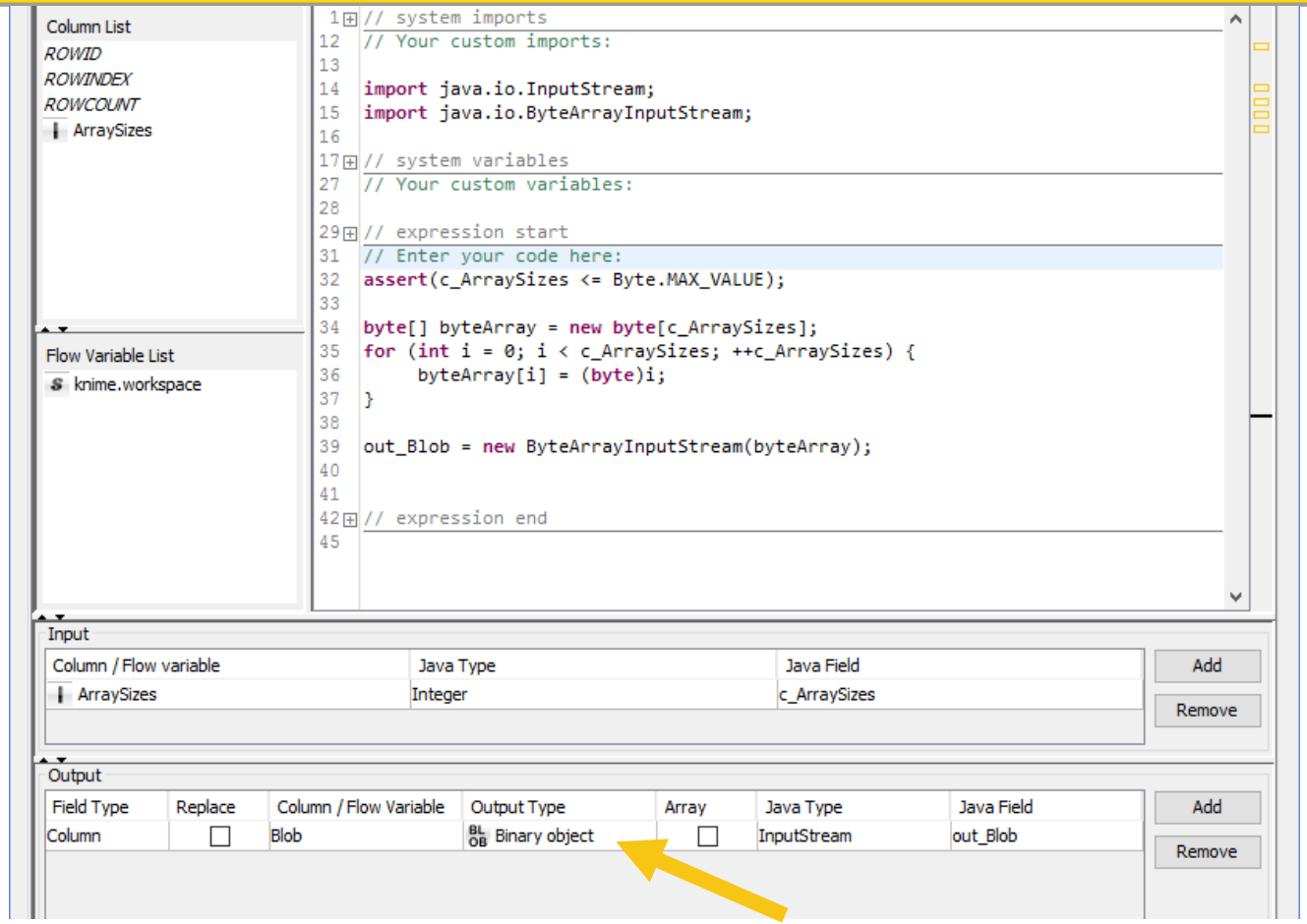
```
public MyDataCell createFromFoo(Foo source) {
     return new MyDataCell(source);
}
```

Custom Type Support in Java Snippets – Annotations

In an existing DataCellFactory implementation:

```
@FactoryMethod // <-
public MyDataCell createFromFoo(Foo source) {
     return new MyDataCell(source);
}
```

Support for Existing DataTypes



Extendibility

- A same/similar API for Python and R could possibly happen
- Current focus lies on **Java** Snippets