```
#!/usr/bin/python3.4
# -*-coding:Utf-8 -*

from lib.inherit_docstring import inherit_docstring
from lib.choices import choices
from lib.gauss_int import gauss_int
from random import randint

from src.meta.ABCInheritableDocstringsMeta import ABCInheritableDocstringsMeta
from mario.bridge.events.action_events import Jump, Left, Right, Down, Fire
from src.EvolutiveGenerator.GeneticElementFactory import GeneticElementFactory
from src.entities.ActionEventData import ActionEventData


class ActionEventDataFactory(GeneticElementFactory, metaclass=ABCInheritableDocstringsMeta):
    """ActionEventData factory """

    @property
    @inherit_docstring
    def genetic_element_class(self):
        return ActionEventData

    ACTION_CLASSES = (Jump, Left, Right, Down, Fire)


    @classmethod
    @inherit_docstring
    def create(cls):
        action_class = cls.createActionClass()
        return ActionEventData(action_class, cls.createDuration(action_class))

    @classmethod
    @inherit_docstring
    def mutate(cls, element):
        if randint(0, 1):
            element.action_class = cls.createActionClass()
        else:
            element.duration = cls.createDuration(element.action_class)


    @classmethod
    def hydrate(cls, data):
        for action_class in cls.ACTION_CLASSES:
            if action_class.__name__ == data['action_class']:
                return ActionEventData(action_class, data['duration'])
        return ValueError("Action class {} doesn't exist.".format(data['action_class']))


    @classmethod
    def createActionClass(cls):
        return choices(cls.ACTION_CLASSES, weights=[35, 10, 35, 10, 10])[0]

    @staticmethod
    def createDuration(action_class):
        if action_class == Jump:
            return gauss_int(2, 38)
        else:
            return randint(0, 25)
```