UML

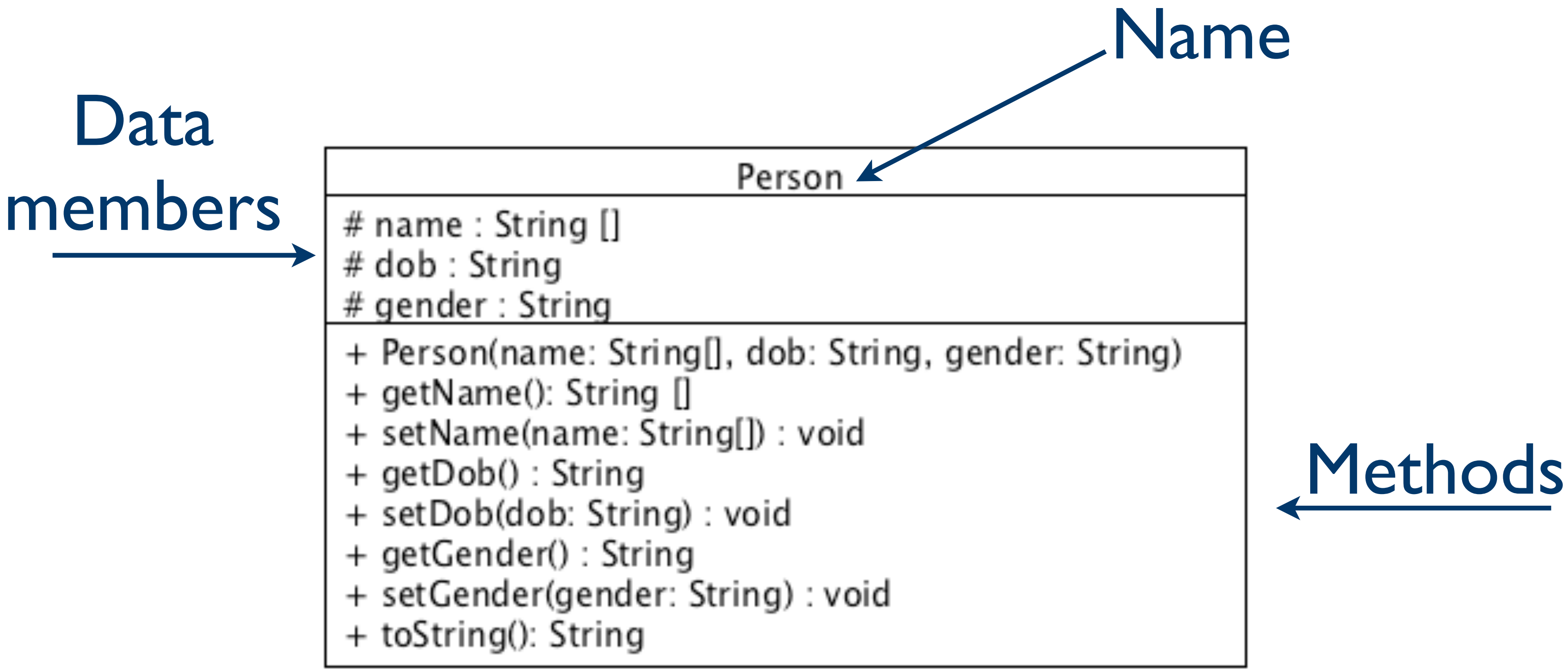
Unified Modeling Language (UML) allows us to express the design of a program before writing any code.

It is language-independent.

An extremely expressive language.

We’ll use only a small part of the language, Class Diagrams, to represent basic OO design.

Example: Class Person



Name

Data members

Methods

```
Person
# name : String []
# dob : String
# gender : String
+ Person(name: String[], dob: String, gender: String)
+ getName(): String []
+ setName(name: String[]) : void
+ getDob() : String
+ setDob(dob: String) : void
+ getGender() : String
+ setGender(gender: String) : void
+ toString(): String
```

Notation

Data members:

```
name: type
```

Methods:

```
methodName(param1: type1, param2: type2,... ): returnType
```

Visibility:

– private

+ public

# protected

~ package

Static: underline

Conventions

Make all non-final instance variables either:

- *private*: accessible only within the class, or
- *protected*: accessible only within the package.

When desired, give outside access using “getter” and “setter” methods.

[A *final* variable cannot change value; it is a constant.]

Access Modifiers

Classes can be declared public or package-private.

Members of classes can be declared public, protected, package-protected, or private.

Modifier	Class	Package	Subclass	World
`public`	Yes	Yes	Yes	Yes
`protected`	Yes	Yes	Yes	No
`default (package private)`	Yes	Yes	No	No
`private`	Yes	No	No	No