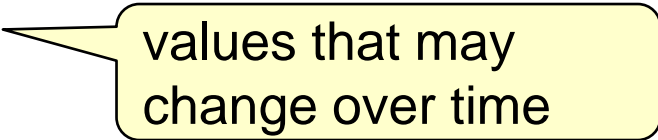

Class/object example: Fields

COMP 102

Victoria University of Wellington

Remembering state

- Each CartoonCharacter object must remember:
 - its state:
 - position
 - emotion
 - direction
 - the folder of image files that it is using.
 - its size
- Can't be stored in local variables in a method
 - local variables are “lost” when the method finishes.
- Have to be stored in the Object itself
 - ⇒ **fields**



values that may change over time

CartoonCharacter Objects

- Objects need places to store values – called “Fields”

CartoonCharacter-24

| | | | |
|------------|----------------------|--------------|----------------------|
| figX: | <input type="text"/> | imagePrefix: | <input type="text"/> |
| figY: | <input type="text"/> | wd: | <input type="text"/> |
| emotion: | <input type="text"/> | ht: | <input type="text"/> |
| direction: | <input type="text"/> | | |

CartoonCharacter-27

| | | | |
|------------|----------------------|--------------|----------------------|
| figX: | <input type="text"/> | imagePrefix: | <input type="text"/> |
| figY: | <input type="text"/> | wd: | <input type="text"/> |
| emotion: | <input type="text"/> | ht: | <input type="text"/> |
| direction: | <input type="text"/> | | |

Using fields:

A method can refer to a field of the object it was called on:

this . *fieldname*

eg:

note: fields have no ()

```
public void lookLeft( ) {
    this.erase( ) ;
    this.direction = "left";
    this.draw( ) ;
}
```

Object the method
was called on:
E.g. cf1.lookLeft();

```
public void draw( ) {
    String filename = this.imagePrefix + "-" + this.direction + "-" +
        this.emotion + ".png" ;
    UI.drawImage(filename, this.figX, this.figY, this.wd, this.ht);
    UI.sleep(500); // wait 500 mS
}
```

Using fields:

Object

CartoonCharacter-24

| | | | |
|------------|---------|--------------|-------|
| figX: | 150 | wd: | 40 |
| figY: | 300 | ht: | 80 |
| emotion: | "smile" | imagePrefix: | "jim" |
| direction: | "right" | | |

```
cf1. lookLeft( );
cf1. walk(20);
```

cf1: CartoonCharacter-24

ID of Object

Method worksheet

```
public void lookLeft( ) {
```

this: CartoonCharacter-

```
    this.erase( );
    this.direction = "left";
    this.draw( );
}
```



Using fields:

Object

CartoonCharacter-24

| | | | |
|------------|---------|--------------|-------|
| figX: | 150 | wd: | 40 |
| figY: | 300 | ht: | 80 |
| emotion: | "smile" | imagePrefix: | "jim" |
| direction: | "left" | | |

```

:
✓ cf1.lookLeft( );
  cf1.walk(20);
:

```

cf1: CartoonCharacter-24

```

public void lookLeft( ) {

```

this: CartoonCharacter-24

```

  ✓ this.erase( ) ;
  ✓ this.direction = "left";
  ✓ this.draw( ) ;
}

```



Using fields:

Object

CartoonCharacter-24

| | | | |
|------------|---------|--------------|-------|
| figX: | 150 | wd: | 40 |
| figY: | 300 | ht: | 80 |
| emotion: | "smile" | imagePrefix: | "jim" |
| direction: | "left" | | |

✓ cf1.lookLeft();
 cf1.walk(20);
 :

cf1: CartoonCharacter-24

```
public void walk (double dist) {
```

this: CartoonCharacter-

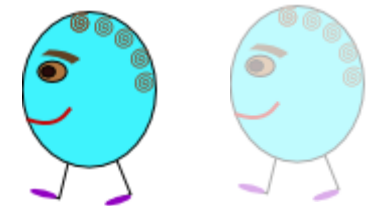
```
    this.erase( );
```

```
    if ( this.direction.equals("right")) { this.figX = this.figX + dist; }
```

```
    else { this.figX = this.figX - dist; }
```

```
    this.draw( );
```

```
}
```



Objects and Classes

Classes define objects:

- Fields: places in an object that store the information associated with the object
methods can refer to fields of the object they were called on:
`this.fieldname`

How do you set up the fields?

- Methods: can be called on any object of the class
- Constructors: specify how to set up an object when it is first created.
- Constants: specify names for values

Setting up an object

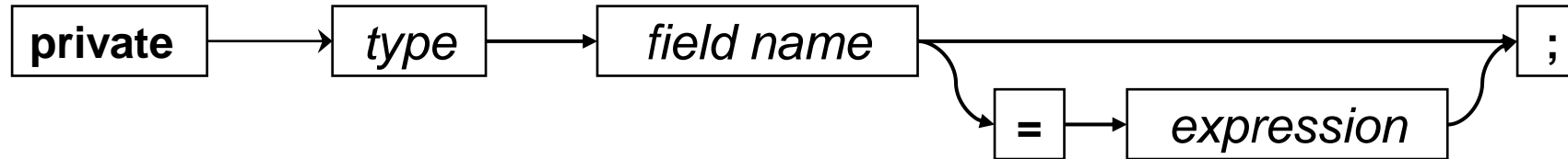
Must declare the Fields of an object

- Declared in the class
(not inside a method)
- Must specify the type and the name
(just like local variables in methods)
- Can specify an initial value (but you don't have to!)
if not, automatically initialised with 0 or null
(unlike local variables)
- Have a visibility specifier (“**private**”)
- Fields remain indefinitely
(unlike local variables)
- The set of field declarations is a template for the object
(just like a method is a template for a worksheet).



Just like local variables
must be declared

Syntax of Field declarations:



```
public class CartoonCharacter {
```

```
// fields
```

```
private double figX;
```

```
private double figY;
```

```
private String direction = "right";
```

```
private String emotion = "smile";
```

```
private String imagePrefix;
```

```
private double wd = 40;
```

```
private double ht=80;
```

```
// methods .....
```

Like variables, BUT
 (a) NOT inside a method
 (b) have **private** in front

```
// current position of character
```

```
// current direction it is facing
```

```
// current emotion
```

```
// base name of images
```

```
// dimensions of figure
```

Every CartoonCharacter will start with these values in the fields.