Method calls

A class is also a type

**Patient**: a type (as well as a class)

**Values**: names appearing in tabs of objects

**Operations**: none

```
function call:       p . getName()

procedure call:     p . deposit(250.00) ;
```

General form:

```
<variable-name> .  <method-name> ( … ) ;
```
Referencing fields

Patient

<table>
<thead>
<tr>
<th>name</th>
<th>Jack Smith</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>23 Spruce, ...</td>
</tr>
<tr>
<td>owes</td>
<td>$250.00</td>
</tr>
</tbody>
</table>

**reference field name:** p . name

**Example of use:**

```
s = p.name;
```

**reference field name:** p . owes

**Example of use:**

```
p.owes = p.owes - 250;
```

Common practice is *not* to reference fields but to use only the methods of p. Fields can be made “private” so they cannot be accessed this way. We discuss the reasons for this later.
The value null

null means the absence of an object name.

You can’t reference a component (method or field) of a non-existent object.
Creating an object

```java
Patient p;

p = new Patient();
```

**Evaluation of new Patient():**
1. Create new manila folder of class Patient
2. Yield as value the name on the tab of the new folder.

```java
Patient p;

p = new Patient();
```
1. Referencing a component of a manila folder (object) whose name is in a variable.
   
   p.getName()
   p.deposit(250);
   p.owes

2. Value null in a variable means the absence of the name of an object.

3. To evaluate new Patient()
   (a) Create (draw) a folder of class Patient;
   (b) Yield name of folder as the result.