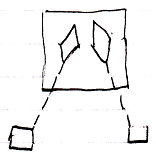
|  |
| --- |
| Circle Language Spec: Commands |

## Commands Example Diagrams

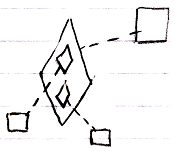
This article attempts to systematically demonstrate different situations of commands.

### Command Calls in a Command Definition



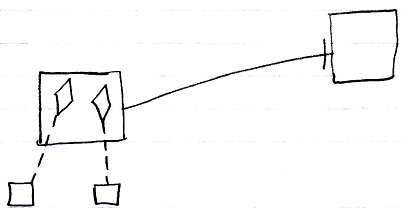
The large square is the parent command definition. The diamonds inside the parent command are two command calls. The diamonds are connected to squares with dashed lines. This makes those squares the command calls’ command definitions.

### Command Calls in a Command Call



The bigger diamond is a command call, because it is tied to its definition, the larger square, with a dashed line, a class line. The smaller diamonds inside the larger diamonds are command calls inside a command call. The smaller diamonds also point out their definitions, the smaller squares, with a dashed line, a class line.

### Command Calls in a Command Reference

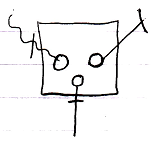


The square on the left is a command reference, because it points out its target definition, the square on the right, with a solid line, an object line. The command reference contains two command calls, displayed as diamonds. Those diamonds are calls, because they point out their definitions, the smaller squares at the bottom, with class lines, dashed lines. The square on the right could also have displayed the same calls, because both larger squares represent the same command object, that have the same contents.

The command reference and its target could also be replaced with a diamond arbitrarily, which make the command reference or target command executable.

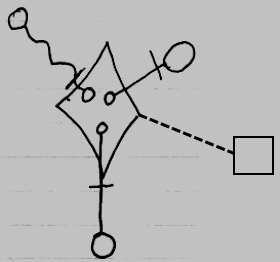
### Parameters in a Command Definition

The diagram below displays an inactive command definition with parameters:



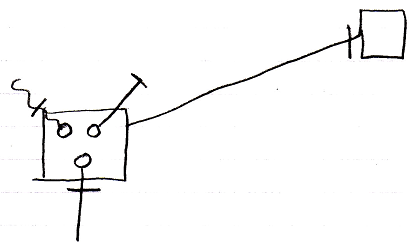
The square is a command definition. The circles inside of it are the parameters. The rest of what’s visible in the diagram are the parameter passings. Parameter passings are explained in the *parameter* articles. The parameter passings are the lines connected to the circles, that are marked with an access mark.

### Parameters in a Command Call



The diamond shape is a command call. What makes it a command call, is that it is tied to its command definition, the square, with a class line, the dashed line. The circles inside the diamond are the parameters. The parameters are tied to objects on the outside with parameter passings. The parameter passings are the lines connecting the smaller circles inside the diamond, the parameters, to the larger circles ouside the diamond. Parameter passings are explained later in the *parameter* articles.

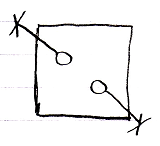
### Parameters in a Command Reference



The larger square on the left is a command reference. It is tied to its reference target, the square on the right, with an object line, which is a solid line. The circles inside the square on the left are the parameters of the command reference. The lines tied to those are the parameter passing types. The parameter passings are the lines connected to the circles, that are marked with an access mark. The square on the right could have had the same contents as the square on the left, because both command reference and command definition represent the same command definition.

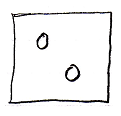
Either of the squares could have been replaced by a diamond, making the command symbol executable, rather than dormant.

### Private Objects in a Command Definition



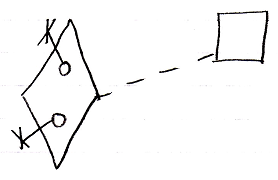
The square is a command definition. The circles inside of it are private objects. In diagram above the fact that they are private is displayed with the object line and the cross it ends with indicating, that you can not reference these objects from the outside.

The following picture is another possible expression of the private objects:



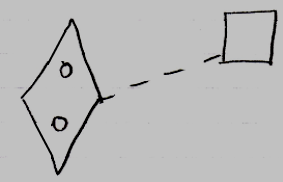
Because public object, parameters, always come with an indication of the parameter passing, you may not need to express the objects’ being private, because it would be obvious: if they do not have a parameter passing, then they are private.

### Private Objects in a Command Call



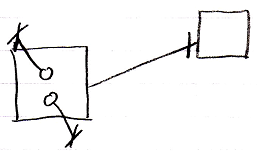
The diamond is a command call. Its definition is indicated with the dashed line connecting it to the square, which is the definition. The circles inside the diamond are the private objects of the command call. The fact, that they are private is displayed with the object line and the cross it ends with indicating, that you can not reference these objects from the outside.

Another possible expression of the private objects is the following:



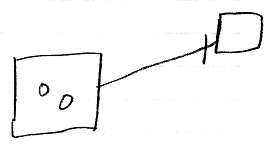
Because public object, parameters, always come with an indication of the parameter passing, you may not need to express the objects’ being private, because it would be obvious: if they do not have a parameter passing, then they are private.

### Private Objects in a Command Reference



The square on the left is a command reference. Its target definition is indicated with a solid line, which is an object, pointing at the square on the right, which is the target definition. The circles inside the square on the left are the private objects of the command call. The fact, that they are private is displayed with the object line and the cross it ends with indicating, that you can not reference these objects from the outside.

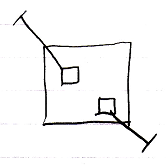
Another possible expression of the private objects is the following:



Because public object, parameters, always come with an indication of the parameter passing, you may not need to express the objects’ being private, because it would be obvious: if they do not have a parameter passing, then they are private.

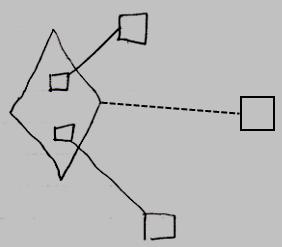
Either of the square could have been replaced by a diamond, making the command symbol executable, rather than dormant.

### Command References in a Command Definition



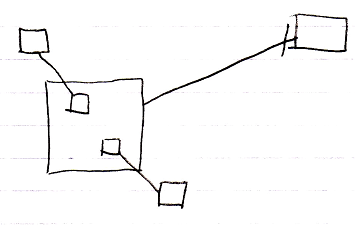
The larger square is a command definition. The smaller square inside them are command references in a command definition. They are actually *public* command references. This makes the command references *parameters* of the command definition. The command references have object lines, solid ones, going outside, ending at an access mark. Those object lines are indications of parameter passing, indicating that the command references are so called Reference parameters. Parameter passings are explained in the *parameter* articles.

### Command References in a Command Call



The diamond is a command call. The command call’s definition is indicated with the dashed line, pointing to the square on the right, which is the command definition. The smaller squares inside the call are command references in a command call. They are actually *public* command references. This makes the command references *parameters* of the command call. The command references have object lines, solid ones, going outside, ending at the targets of the command references.

### Command References in a Command Reference



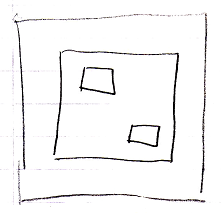
The larger square on the left is a command reference. The target of the command reference is the square on the right, a command definition. The larger square on the left, contains again two command references, smaller squares, that are connected to their reference targets on the outside of the larger command reference.

Most of the squares could have been replaced by diamond shapes, making the command symbol executable, rather than dormant. But the smaller squares inside the larger square can not be replaced by diamonds, because a command can not contain active command references, for reasons explained by the article *Parent Controls Its Sub-Executions*.

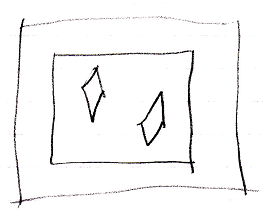
### Clauses in Clauses

Because clauses are always part of a parent command, the parent command is displayed in the pictures below as the outer square. Inside the square is a clause, and inside that symbol are two clauses in a clause. Inactive clauses are squares. Active clauses are diamonds.

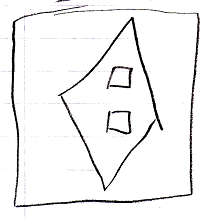
Inactive clauses in an inactive clause in a command definition:



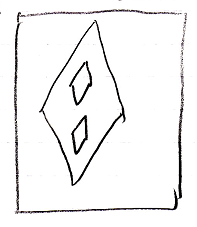
Active clauses in an inactive clause in a command definition:



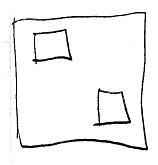
Inactive clauses in an active clause in a command definition:



Active clauses in an active clause in a command definition:

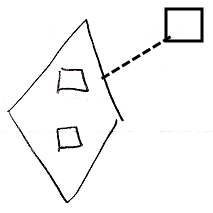


### Inactive Clauses in a Command Definition



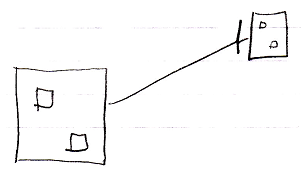
The larger square is a command definition. The smaller squares inside the larger square are inactive clauses inside a command definition. Squares inside a square, that do not redirect, are inactive clauses.

### Inactive Clauses in a Command Call



The diamond shape is a command call. The command definition of the command call is pointed out by the dashed line, pointing at the square on the right, which is the command definition. The diamond contains two squares, that do not redirect. Those are the inactive clauses inside the command call. They are clauses, because they are situated inside a command symbol and they do not redirect their target.

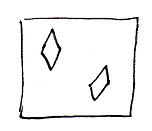
### Inactive Clauses in a Command Reference



The large square on the left is a command reference. It is pointing out its target command definition with an object line, the solid line, targeting the square on the right, which is the command definition. The command reference on the left contains two inactive clauses, smaller squares, that do not redirect their target. The target definition on the right also shows the smaller squares inside of it. Both command reference and target definition both represent the same command object, so their contents are also the same, so they both show inactive clauses.

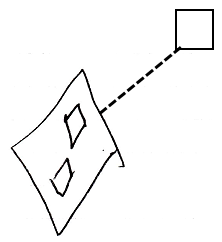
Either of the larger squares could have been replaced by a diamond, making the command symbol executable, rather than dormant.

### Active Clauses in a Command Definition



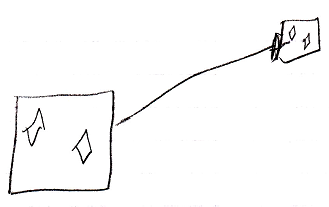
The square is a command definition. The diamond shapes squares inside the square are active clauses inside a command definition. Diamonds inside a square, that do not redirect, are active clauses.

### Active Clauses in a Command Call



The diamond shape is a command call. The command definition of the command call is pointed out by the dashed line, pointing at the square, which is the command definition. The diamond contains two diamond shapes, that do not redirect. Those are the active clauses inside the command call. They are clauses, because they are situated inside a command symbol and they do not redirect their target.

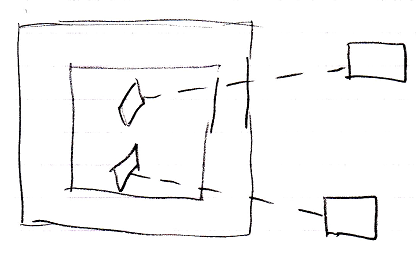
### Active Clauses in a Command Reference



The large square on the left is a command reference. It is pointing out its target command definition with an object line, the solid line, targeting the square on the right, which is the command definition. The command reference on the left contains two active clauses, diamonds, that do not redirect their target. The target definition on the right also shows the diamonds inside of it. Both command reference and target definition both represent the same command object, so their contents are also the same, so they both show active clauses.

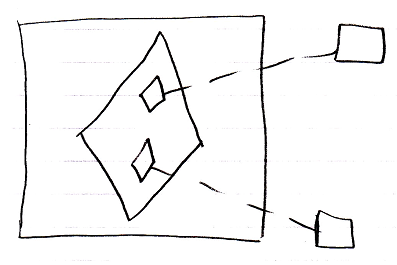
Either of the squares could have been replaced by a diamond, making the command symbol executable, rather than dormant.

### Command Calls in an Inactive Clause



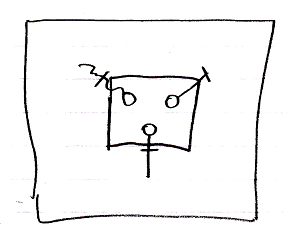
The large square is the command definition the clause is in. The square inside of it, is the inactive clause. It contains two diamonds, which are the command calls inside the inactive clause. The diamonds redirect their command definition with a class line, a dashed line, connected to the squares on the right, which are the command definitions, that are called.

### Command Calls in an Active Clause



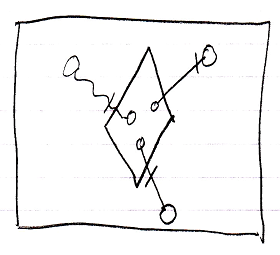
The large square is the command definition the clause is in. The diamond inside of it, is the active clause. It contains two diamonds, which are the command calls inside the active clause. The small diamonds redirect their command definition with a class line, a dashed line, connected to the squares on the right, which are the command definitions, that are called.

### Parameters in an Inactive Clause



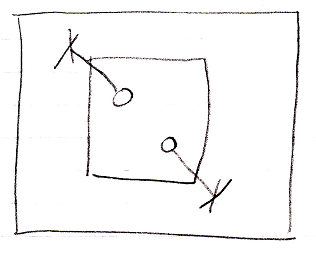
The large square is a command definition the inactive clause is in. The square inside the square is the inactive clause. The inactive clause contains three circles, which are the parameters in the clause. The lines connected to the circles, which are also dissected by an access mark, are the parameter passings. Parameter passings are explained in the *parameter* articles.

### Parameters in an Active Clause



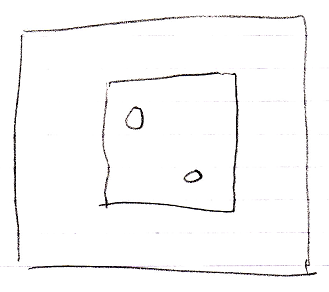
The large square is a command definition the inactive clause is in. The diamond inside the square is the active clause. The diamond contains three circles, which are the parameters in the clause. The circles inside the diamond are connected to the circles outside the diamond with parameter passings. Parameter passings are explained in the *Parameters* articles.

### Private Objects in an Inactive Clause



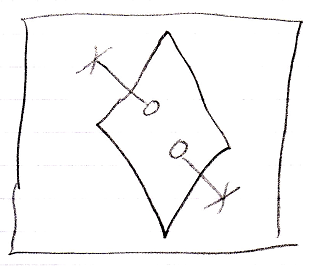
The large square is a command definition, the clause is in. The square inside the larger square is the inactive clause. It contains two circles, which are the private objects of the inactive clause. The lines that go outside, that end at a cross are the indications that the objects are private. They indicate you can not reference the circles from the outside.

Another possible expression of the private objects is the following:



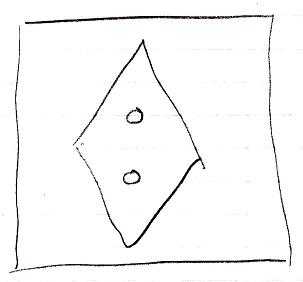
Because public object, parameters, always come with an indication of the parameter passing, you may not need to express the objects’ being private, because it would be obvious: if they do not have a parameter passing, then they are private.

### Private Objects in an Active Clause



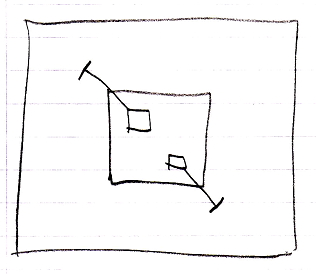
The large square is a command definition, the clause is in. The diamond inside the square is the active clause. It contains two circles, which are the private objects of the active clause. The lines that go outside, that end at a cross are the indications that the objects are private. They indicate you can not reference the circles from the outside.

Another possible expression of the private objects is the following:



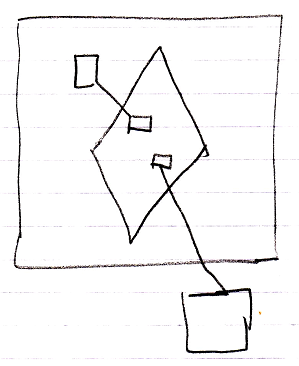
Because public object, parameters, always come with an indication of the parameter passing, you may not need to express the objects’ being private, because it would be obvious: if they do not have a parameter passing, then they are private.

### Command References in an Inactive Clause



The large square is a command definition, the inactive clause is in. The square inside the large square is the inclause. The inactive clause contains two squares, which are the command references. They are actually *public* command references. This makes the command references parameters of the inactive clause. The command references have object lines, solid ones, going outside, ending at an access mark. Those object lines are indications of parameter passing, indicating that the command references are so called Reference parameters. Parameter passings are explained in the *Parameters* articles. The parameters are not filled in, because an inactive clause is a dormant command definition. Only in a call to the dormant command definitions, the parameters are filled in.

### Command References in an Active Clause



The large square is the command definition, that the active clause is in. The diamond is the active clause. The smaller squares inside the diamond are command references in an active clause. They are actually *public* command references. This makes the command references parameters of the active clause. The command references have object lines, solid ones, going outside, ending at the targets of the command references.