INFO213: Lecture 3

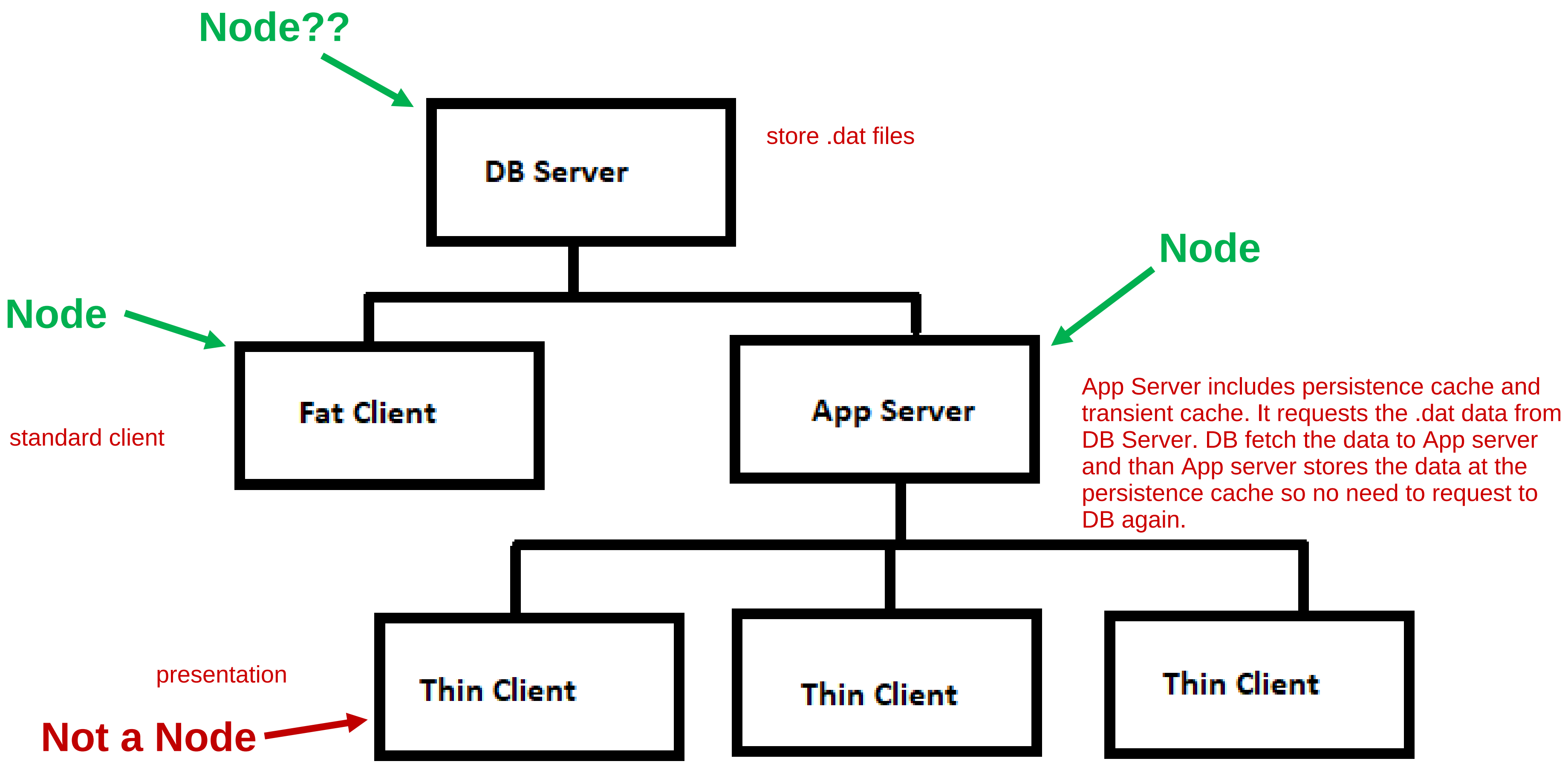
JADE Fundamentals

JADE Nodes, JADE Caching, JADE Syntax, JADE Access Control, JADE Painter

First on the agenda:

- Nodes, Processes and Caches
  - Complicated!
  - Don’t worry, we will keep touching on it...
- Persistent vs Transient objects
- Cache coherency

A JADE System: Find the nodes!



App Server includes persistence cache and transient cache. It requests the .dat data from DB Server. DB fetch the data to App server and than App server stores the data at the persistence cache so no need to request to DB again.

What’s a Node

*A node is a component of a JADE system where **application code** is executed and where **objects are processed**.*

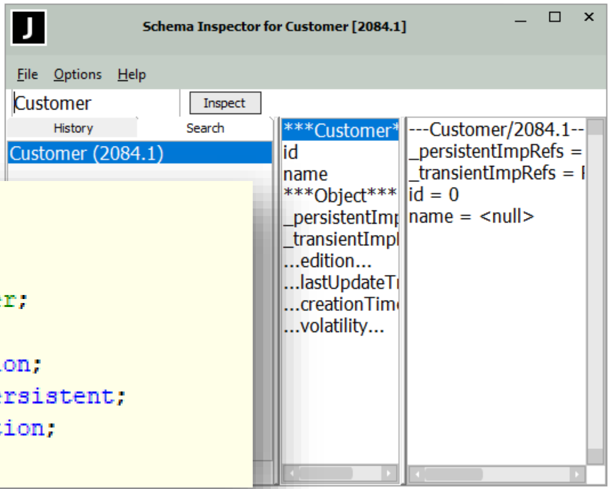
- A node has two caches:
  - A Persistent Cache – Stores *copies* of Persistent Objects in memory store at .dat file
  - A Transient Cache – Stores Transient Objects in memory

short lived, being used and then throw away

- A Node can also run any number of JADE Processes, which are what the application code runs on.

Persistent vs Transient Objects

Persistent


Schema Inspector for Customer [2084.1]

```
***Customer***
id
name
***Object***
_persistentImp
_transientImpl
...edition...
...lastUpdateT
...creationTim
...volatility...
```

```
---Customer/2084.1--
_persistentImpRefs =
_transientImpRefs =
id = 0
name = <null>
```

```
exampleMethod();

vars
    cust : Customer;
begin
    beginTransaction;
    create cust persistent;
    commitTransaction;
end;
```

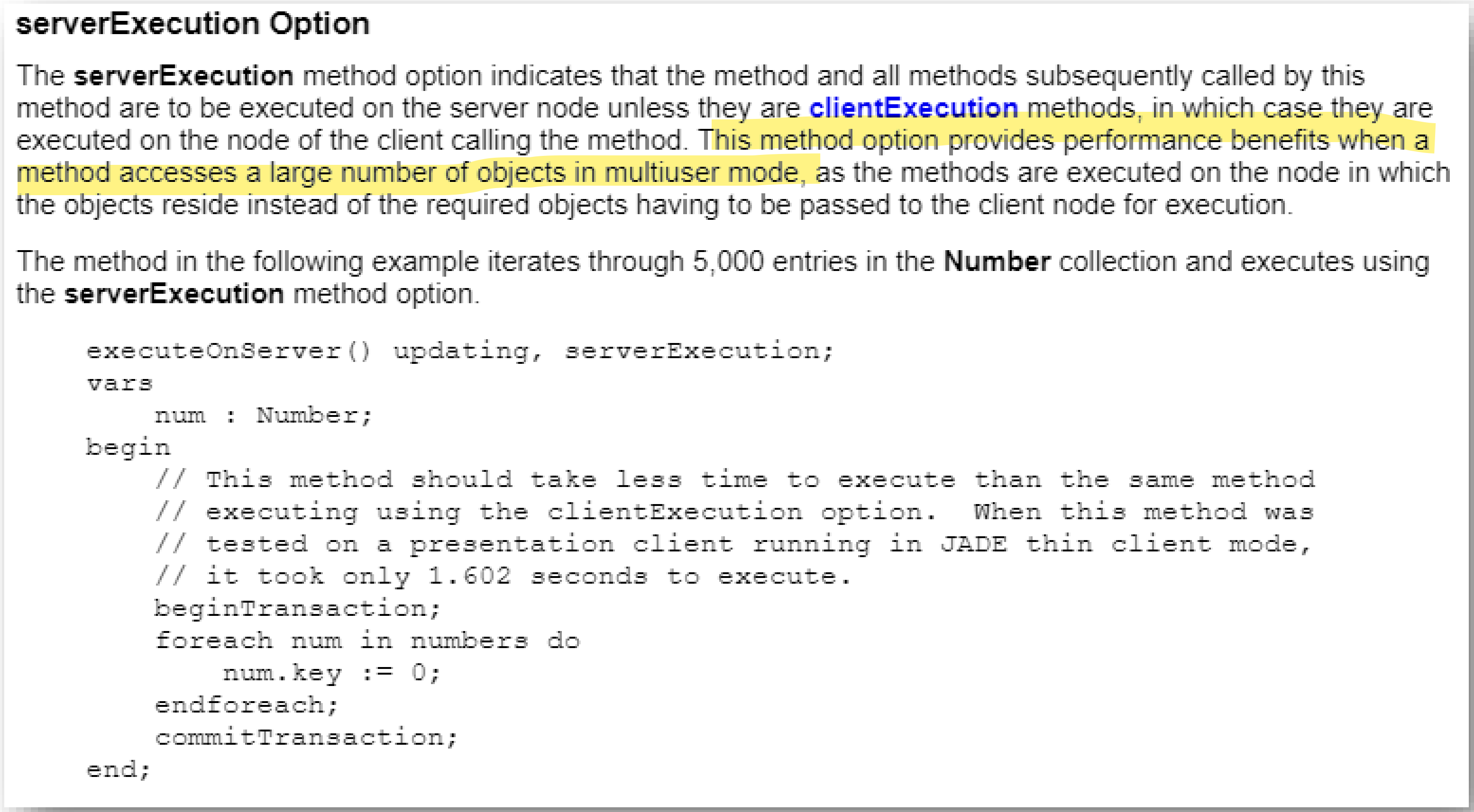
- Persistent Objects are saved to the database
- As such, they are **long-lived**.
- They can be created and modified only inside **beginTransaction; commitTransaction; blocks**

Transient

```
exampleMethod();

vars
    cust : Customer;
begin
    create cust transient;
    // Do something interesting with the customer...

epilog
    delete cust;
end;
```

- Transient Objects are stored in memory
- As such, they are **short-lived**.
- You do not need any transaction for them, but you SHOULD delete them when you're done with them.

Server Execution



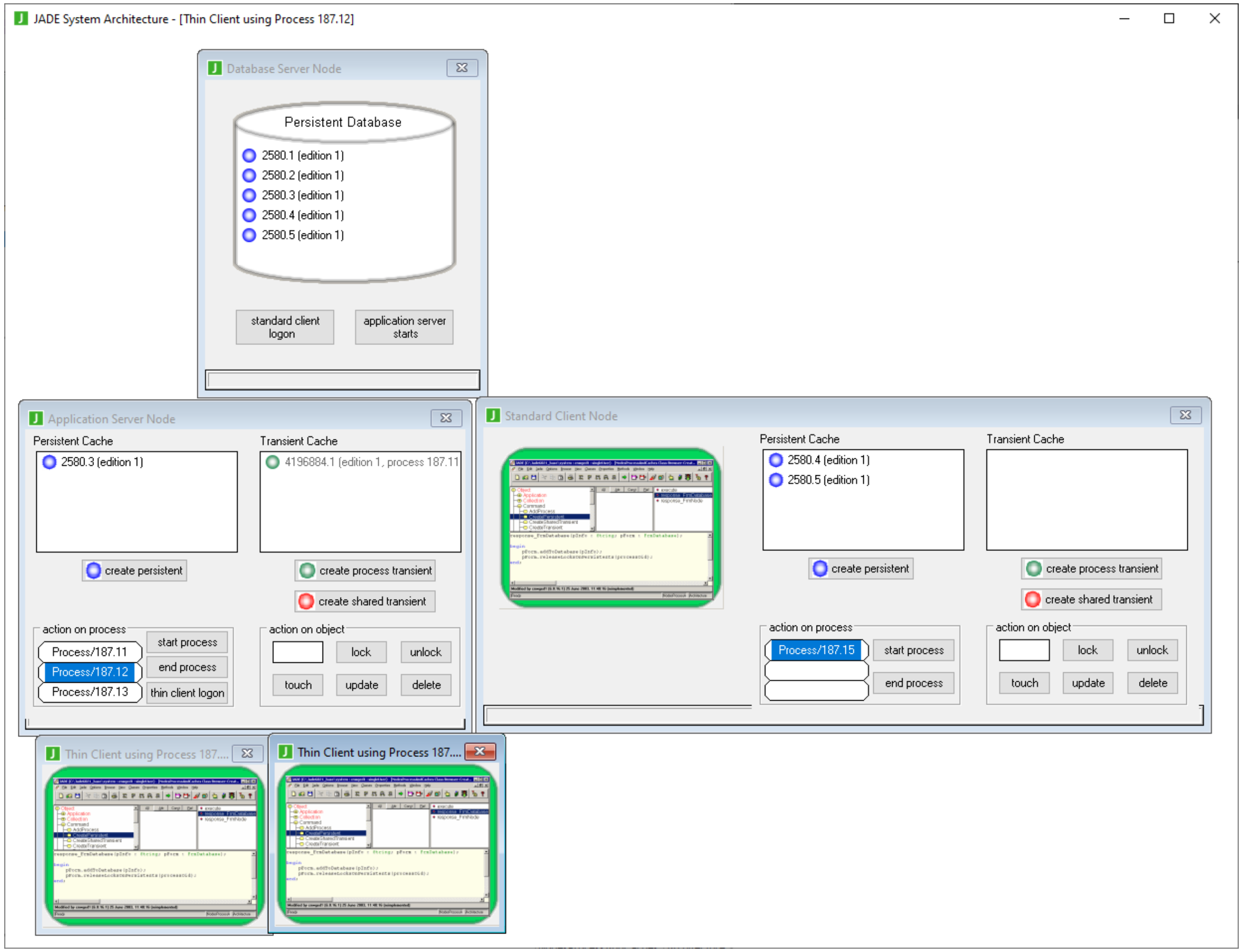
serverExecution Option

The **serverExecution** method option indicates that the method and all methods subsequently called by this method are to be executed on the server node unless they are **clientExecution** methods, in which case they are executed on the node of the client calling the method. This method option provides performance benefits when a method accesses a large number of objects in multiuser mode, as the methods are executed on the node in which the objects reside instead of the required objects having to be passed to the client node for execution.

The method in the following example iterates through 5,000 entries in the **Number** collection and executes using the **serverExecution** method option.

```
executeOnServer() updating, serverExecution;
vars
    num : Number;
begin
    // This method should take less time to execute than the same method
    // executing using the clientExecution option.  When this method was
    // tested on a presentation client running in JADE thin client mode,
    // it took only 1.602 seconds to execute.
    beginTransaction;
    foreach num in numbers do
        num.key := 0;
    endforeach;
    commitTransaction;
end;
```

Cache Coherency - Demo


JADE System Architecture - [Thin Client using Process 187.12]

Database Server Node

Persistent Database

- 2580.1 (edition 1)
- 2580.2 (edition 1)
- 2580.3 (edition 1)
- 2580.4 (edition 1)
- 2580.5 (edition 1)

Application Server Node

Persistent Cache

- 2580.3 (edition 1)

Transient Cache

- 4196884.1 (edition 1, process 187.11

action on process

- Process/187.11
- Process/187.12
- Process/187.13

Standard Client Node

Persistent Cache

- 2580.4 (edition 1)
- 2580.5 (edition 1)

Transient Cache

action on process

- Process/187.15

Thin Client using Process 187....

Thin Client using Process 187....

JADE Syntax

- Hopefully, you already have already practiced much of this in tutorials!
- We’re going to go through this speedily – can come back these slides for notes if needed!

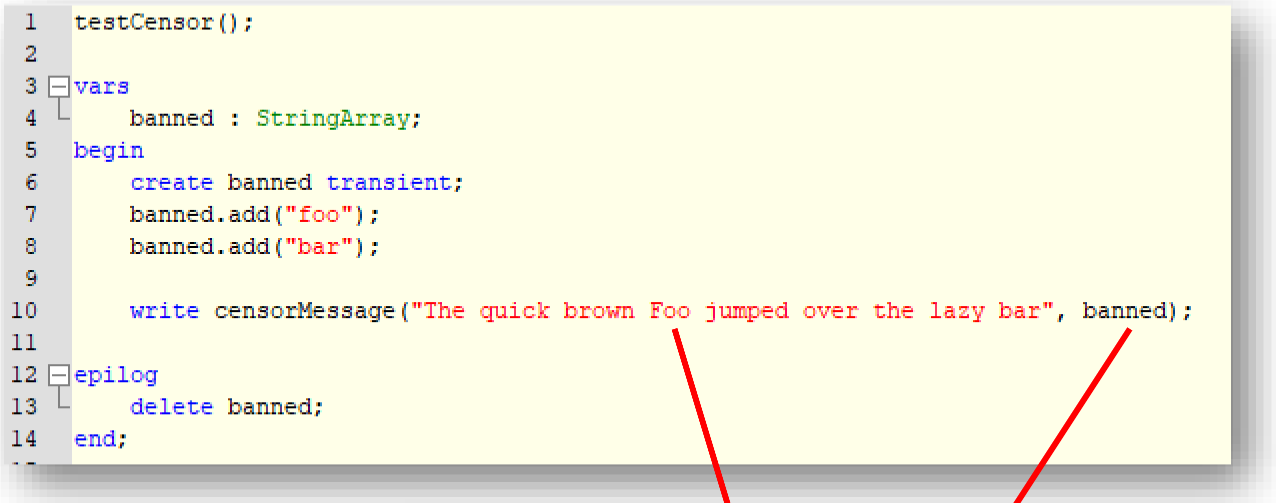
Flow control – If, Foreach, While

- **If** statements control whether code is executed based on a condition (true/false)
- **Foreach** loop performs one block of code multiple times – once for each element in a range or collection
- **While** loop performs one block of code multiple times – as many times as needed to make a condition false
  - Be careful! It's very easy to get stuck in an infinite loop when using these!

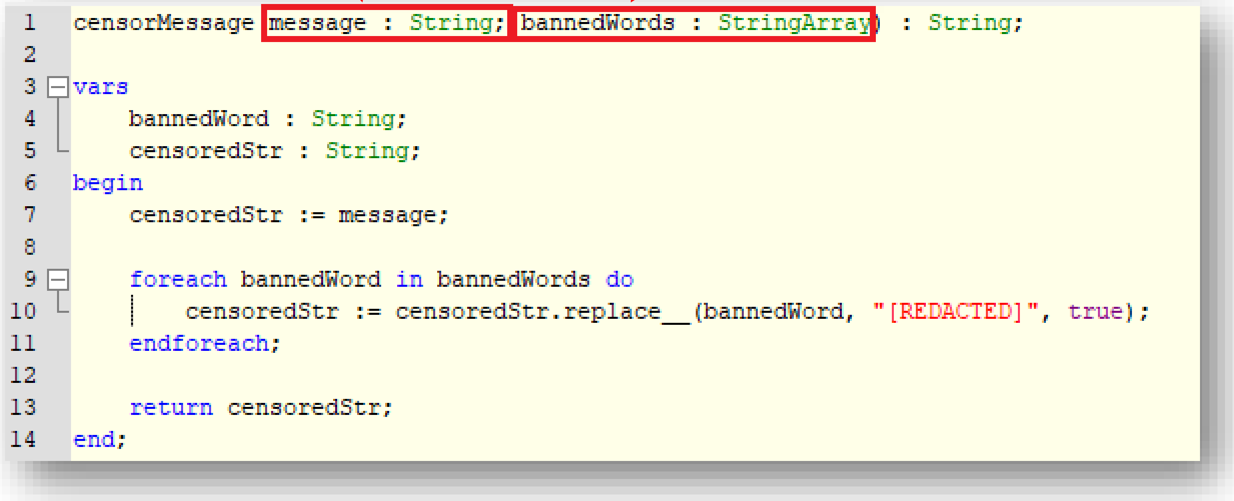
Flow Control - Break and Continue

- Use these while inside a loop (foreach or while)
- **Break** exits out of the loop immediately
- **Continue** merely skips the rest of that iteration of the loop, beginning again at the top of the loop.

Method Parameters

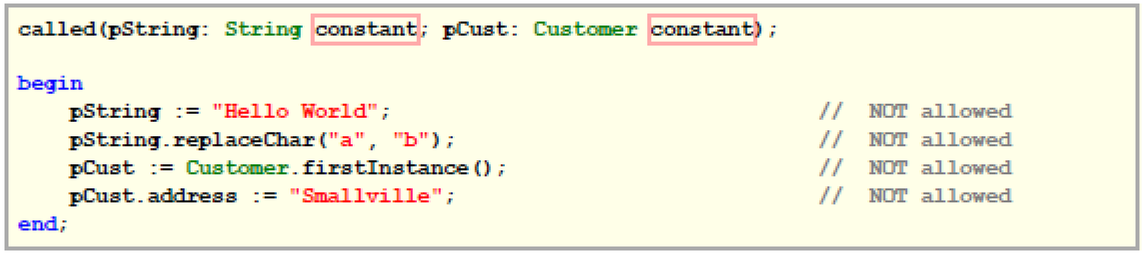


```
testCensor();

vars
    banned : StringArray;
begin
    create banned transient;
    banned.add("foo");
    banned.add("bar");

    write censorMessage("The quick brown Foo jumped over the lazy bar", banned);

epilog
    delete banned;
end;
```

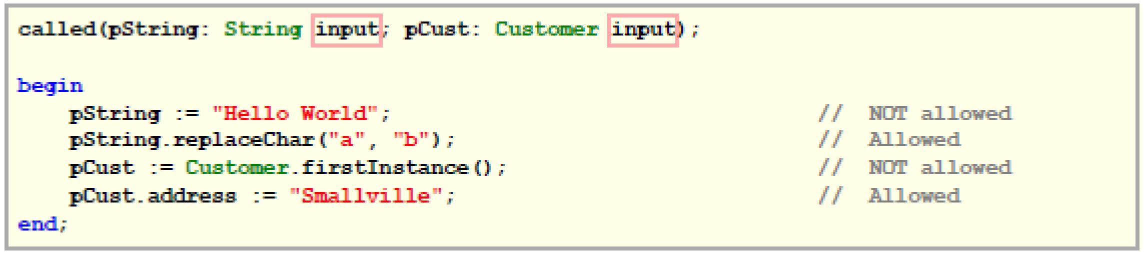


```
censorMessage(message : String; bannedWords : StringArray) : String;

vars
    bannedWord : String;
    censoredStr : String;
begin
    censoredStr := message;

    foreach bannedWord in bannedWords do
        censoredStr := censoredStr.replace__(bannedWord, "[REDACTED]", true);
    endforeach;

    return censoredStr;
end;
```

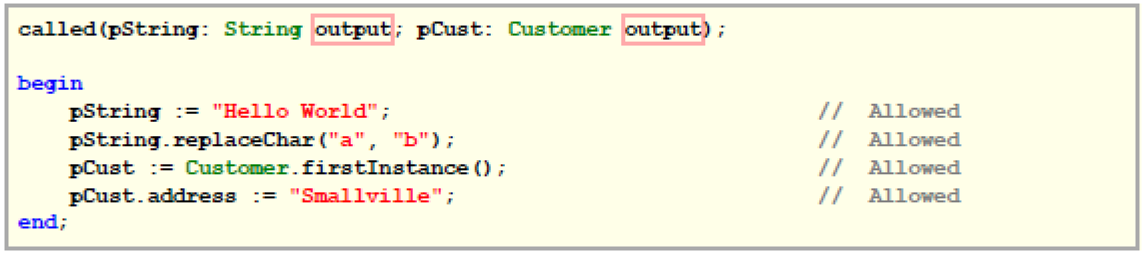
Method Parameters – Usage Options



```
called(pString: String constant; pCust: Customer constant);

begin
    pString := "Hello World";                          //  NOT allowed
    pString.replaceChar("a", "b");                     //  NOT allowed
    pCust := Customer.firstInstance();                 //  NOT allowed
    pCust.address := "Smallville";                     //  NOT allowed
end;
```



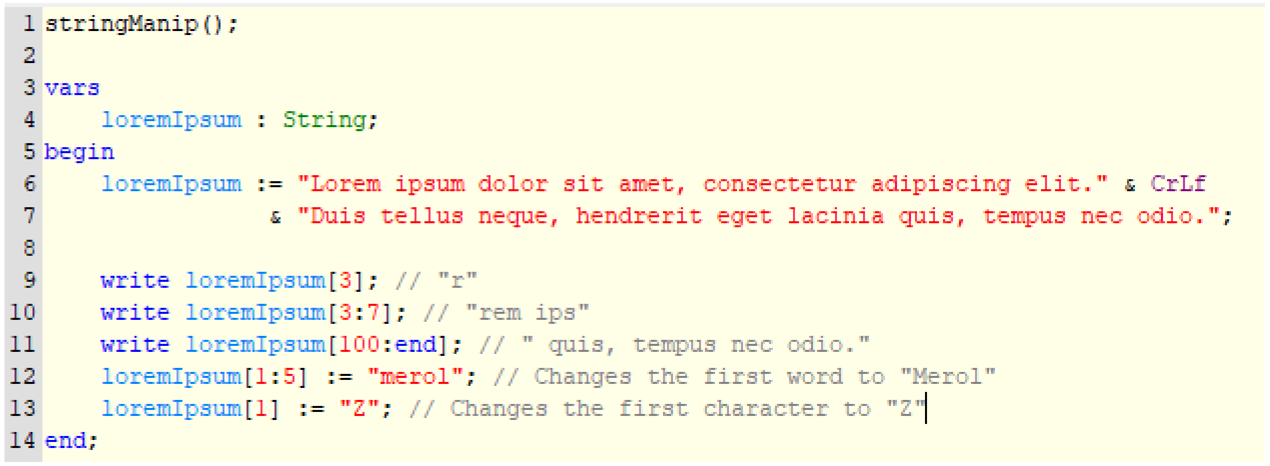
```
called(pString: String input; pCust: Customer input);

begin
    pString := "Hello World";                          //  NOT allowed
    pString.replaceChar("a", "b");                     //  Allowed
    pCust := Customer.firstInstance();                 //  NOT allowed
    pCust.address := "Smallville";                     //  Allowed
end;
```



```
called(pString: String output; pCust: Customer output);

begin
    pString := "Hello World";                          //  Allowed
    pString.replaceChar("a", "b");                     //  Allowed
    pCust := Customer.firstInstance();                 //  Allowed
    pCust.address := "Smallville";                     //  Allowed
end;
```

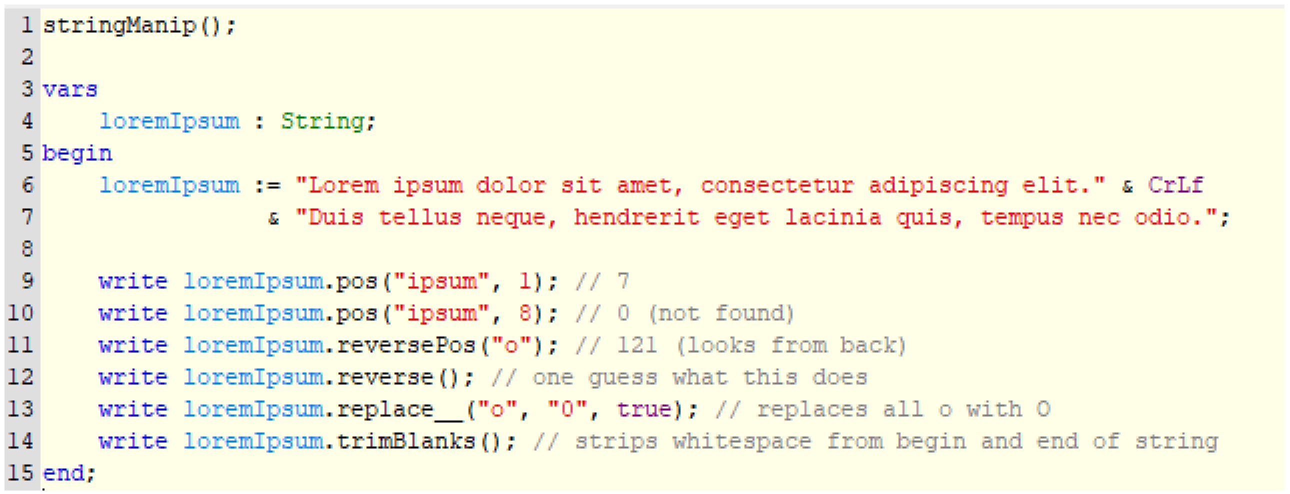
There is also an io option, which is like output but not nulled on entry…

String manipulation : Indexing



```
stringManip();

vars
    loremIpsum : String;
begin
    loremIpsum := "Lorem ipsum dolor sit amet, consectetur adipiscing elit." & CrLf
                & "Duis tellus neque, hendrerit eget lacinia quis, tempus nec odio.";

    write loremIpsum[3]; // "r"
    write loremIpsum[3:7]; // "rem ips"
    write loremIpsum[100:end]; // " quis, tempus nec odio."
    loremIpsum[1:5] := "merol"; // Changes the first word to "Merol"
    loremIpsum[1] := "Z"; // Changes the first character to "Z"
end;
```

String manipulation : Common methods



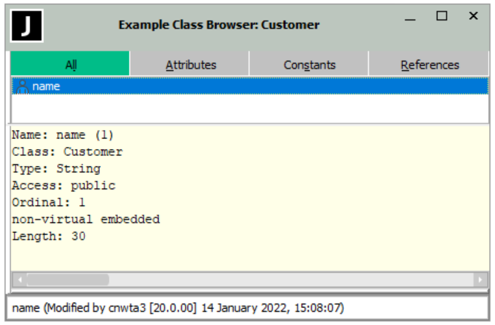
```
stringManip();

vars
    loremIpsum : String;
begin
    loremIpsum := "Lorem ipsum dolor sit amet, consectetur adipiscing elit." & CrLf
                & "Duis tellus neque, hendrerit eget lacinia quis, tempus nec odio.";

    write loremIpsum.pos("ipsum", 1); // 7
    write loremIpsum.pos("ipsum", 8); // 0 (not found)
    write loremIpsum.reversePos("o"); // 121 (looks from back)
    write loremIpsum.reverse(); // one guess what this does
    write loremIpsum.replace__("o", "0", true); // replaces all o with 0
    write loremIpsum.trimBlanks(); // strips whitespace from begin and end of string
end;
```

Encapsulation (Access Control)

- Encapsulation
  - What is a capsule?
  - Why do we put things in capsules?

JADE Property Access

Public



Example Class Browser: Customer

All | Attributes | Constants | References

name

```
Name: name (1)
Class: Customer
Type: String
Access: public
Ordinal: 1
non-virtual embedded
Length: 30
```

name (Modified by cnwta3 [20.0.00] 14 January 2022, 15:08:07)

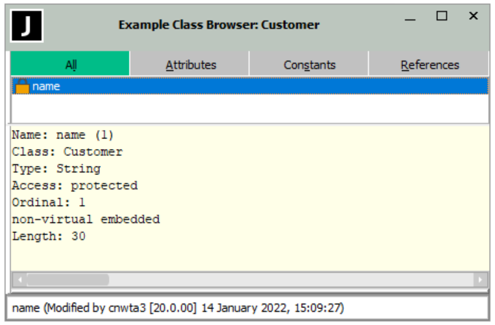
Methods of this class read? ✓

Methods of this class update? ✓

Methods of other classes read? ✓

Methods of other classes update? ✓

Protected



Example Class Browser: Customer

All | Attributes | Constants | References

name

```
Name: name (1)
Class: Customer
Type: String
Access: protected
Ordinal: 1
non-virtual embedded
Length: 30
```

name (Modified by cnwta3 [20.0.00] 14 January 2022, 15:09:27)

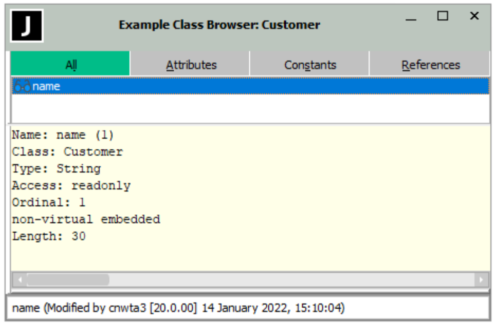
Methods of this class read? ✓

Methods of this class update? ✓

Methods of other classes read? ✗

Methods of other classes update? ✗

Read Only



Example Class Browser: Customer

All | Attributes | Constants | References

name

```
Name: name (1)
Class: Customer
Type: String
Access: readonly
Ordinal: 1
non-virtual embedded
Length: 30
```

name (Modified by cnwta3 [20.0.00] 14 January 2022, 15:10:04)

Methods of this class read? ✓

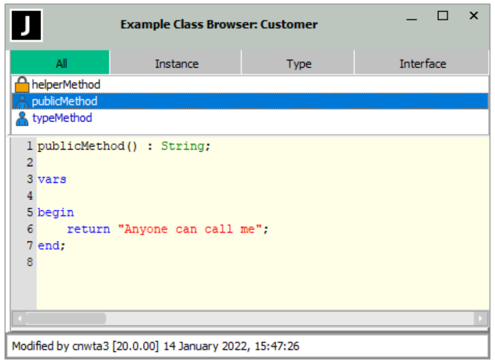
Methods of this class update? ✓

Methods of other classes read? ✓

Methods of other classes update? ✗

JADE Method Access

Public



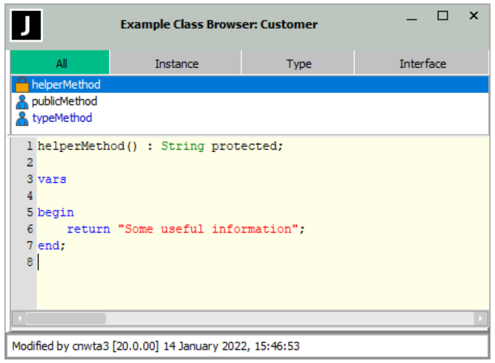
```
publicMethod() : String;

vars

begin
    return "Anyone can call me";
end;
```

Modified by cnwta3 [20.0.00] 14 January 2022, 15:47:26

Part of the class's public interface.

Can be called from any method with an object of this class.

Protected



```
helperMethod() : String protected;

vars

begin
    return "Some useful information";
end;
```

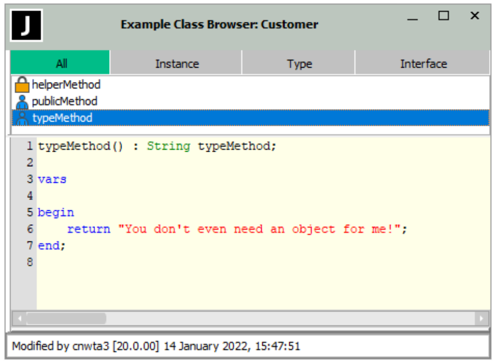
Modified by cnwta3 [20.0.00] 14 January 2022, 15:46:53

Not part of the class’s public interface.

Can only be called from methods of the same object.

Useful for “helper” methods, which split up one longer method into multiple smaller or reuse a common bit of code.

Type Method



```
typeMethod() : String typeMethod;

vars

begin
    return "You don't even need an object for me!";
end;
```

Modified by cnwta3 [20.0.00] 14 January 2022, 15:47:51

Doesn’t require an object of the class to use (can be called directly from the class using ClassName@methodName syntax

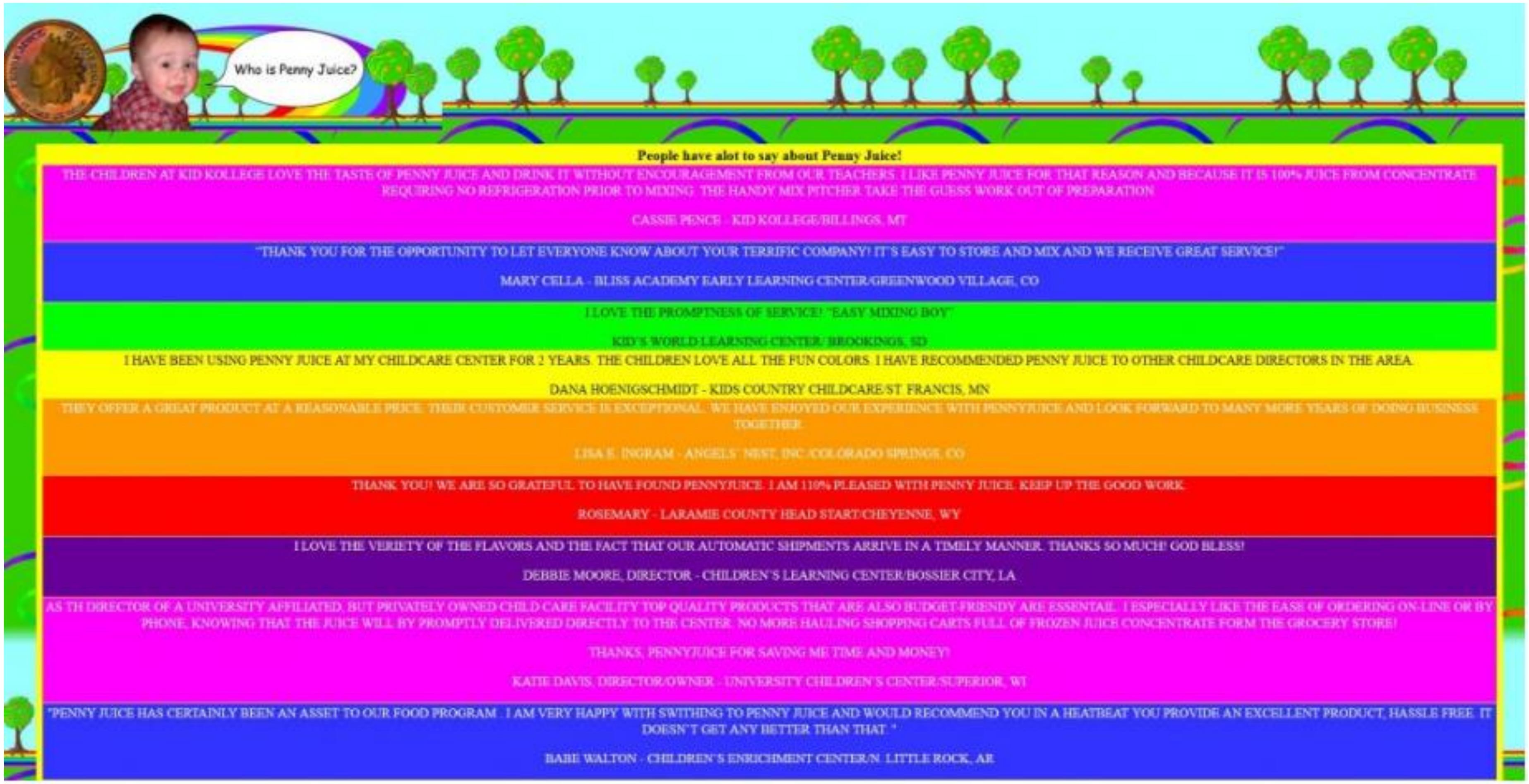
Used for utility methods.

Can’t use any properties or non-type-methods of the class.

GUIs and the JADE Painter

- GUI = Graphical User Interface
- The JADE Painter allows you to design GUIs.
- GUIs aren’t the only sort of interface
  - APIs (Application Programming Interface)
  - CLIs (Command-Line Interface)
- GUIs can be good or bad … How do we measure quality of a GUI?

Is this a good GUI? What's wrong with it?



People have alot to say about Penny Juice!

THE CHILDREN AT KID KOLLEGE LOVE THE TASTE OF PENNY JUICE AND DRINK IT WITHOUT ENCOURAGEMENT FROM OUR TEACHERS. I LIKE PENNY JUICE FOR THAT REASON AND BECAUSE IT IS 100% JUICE FROM CONCENTRATE REQUIRING NO REFRIGERATION PRIOR TO MIXING. THE HANDY MIX PITCHER TAKE THE GUESS WORK OUT OF PREPARATION.

CASSIE PENCE - KID KOLLEGE/BILLINGS, MT

"THANK YOU FOR THE OPPORTUNITY TO LET EVERYONE KNOW ABOUT YOUR TERRIFIC COMPANY! IT'S EASY TO STORE AND MIX AND WE RECEIVE GREAT SERVICE!"

MARY CELLA - BLISS ACADEMY EARLY LEARNING CENTER/GREENWOOD VILLAGE, CO

I LOVE THE PROMPTNESS OF SERVICE! "EASY MIXING BOY"

KID'S WORLD LEARNING CENTER/ BROOKINGS, SD

I HAVE BEEN USING PENNY JUICE AT MY CHILDCARE CENTER FOR 2 YEARS. THE CHILDREN LOVE ALL THE FUN COLORS. I HAVE RECOMMENDED PENNY JUICE TO OTHER CHILDCARE DIRECTORS IN THE AREA.

DANA HOENIGSCHMIDT - KIDS COUNTRY CHILDCARE/ST. FRANCIS, MN

THEY OFFER A GREAT PRODUCT AT A REASONABLE PRICE. THEIR CUSTOMER SERVICE IS EXCEPTIONAL. WE HAVE ENJOYED OUR EXPERIENCE WITH PENNYJUICE AND LOOK FORWARD TO MANY MORE YEARS OF DOING BUSINESS TOGETHER.

LISA E. INGRAM - ANGELS' NEST, INC./COLORADO SPRINGS, CO

THANK YOU! WE ARE SO GRATEFUL TO HAVE FOUND PENNYJUICE. I AM 110% PLEASED WITH PENNY JUICE. KEEP UP THE GOOD WORK.

ROSEMARY - LARAMIE COUNTY HEAD START/CHEYENNE, WY

I LOVE THE VERIETY OF THE FLAVORS AND THE FACT THAT OUR AUTOMATIC SHIPMENTS ARRIVE IN A TIMELY MANNER. THANKS SO MUCH! GOD BLESS!

DEBBIE MOORE, DIRECTOR - CHILDREN'S LEARNING CENTER/BOSSIER CITY, LA

AS TH DIRECTOR OF A UNIVERSITY AFFILIATED, BUT PRIVATELY OWNED CHILD CARE FACILITY TOP QUALITY PRODUCTS THAT ARE ALSO BUDGET-FRIENDY ARE ESSENTAIL. I ESPECIALLY LIKE THE EASE OF ORDERING ON-LINE OR BY PHONE, KNOWING THAT THE JUICE WILL BY PROMPTLY DELIVERED DIRECTLY TO THE CENTER. NO MORE HAULING SHOPPING CARTS FULL OF FROZEN JUICE CONCENTRATE FORM THE GROCERY STORE!

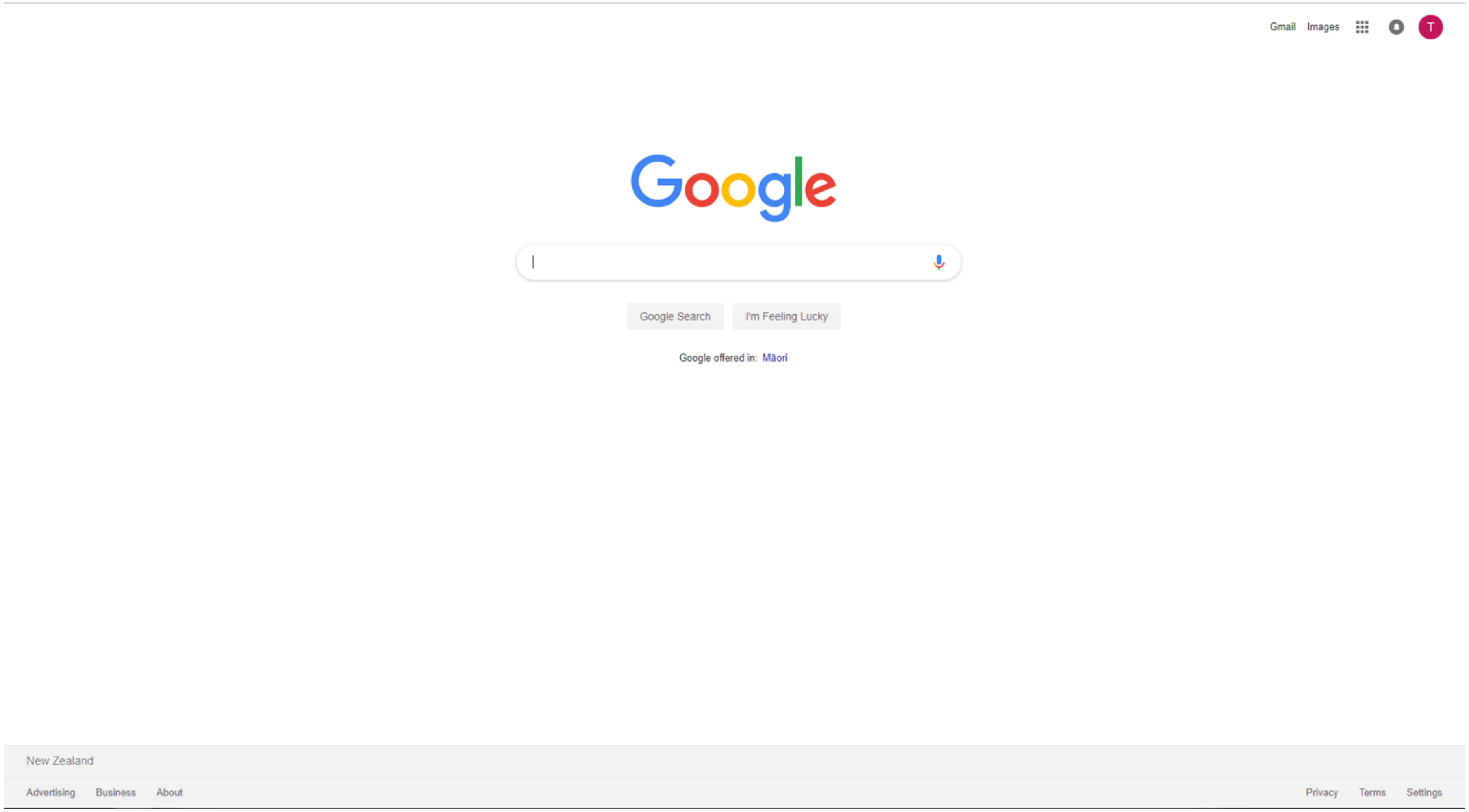
THANKS, PENNYJUICE FOR SAVING ME TIME AND MONEY!

KATIE DAVIS, DIRECTOR/OWNER - UNIVERSITY CHILDREN'S CENTER/SUPERIOR, WI

"PENNY JUICE HAS CERTAINLY BEEN AN ASSET TO OUR FOOD PROGRAM . I AM VERY HAPPY WITH SWITHING TO PENNY JUICE AND WOULD RECOMMEND YOU IN A HEATBEAT YOU PROVIDE AN EXCELLENT PRODUCT, HASSLE FREE. IT DOESN'T GET ANY BETTER THAN THAT."

BABE WALTON - CHILDREN'S ENRICHMENT CENTER/N. LITTLE ROCK, AR

How about this one?



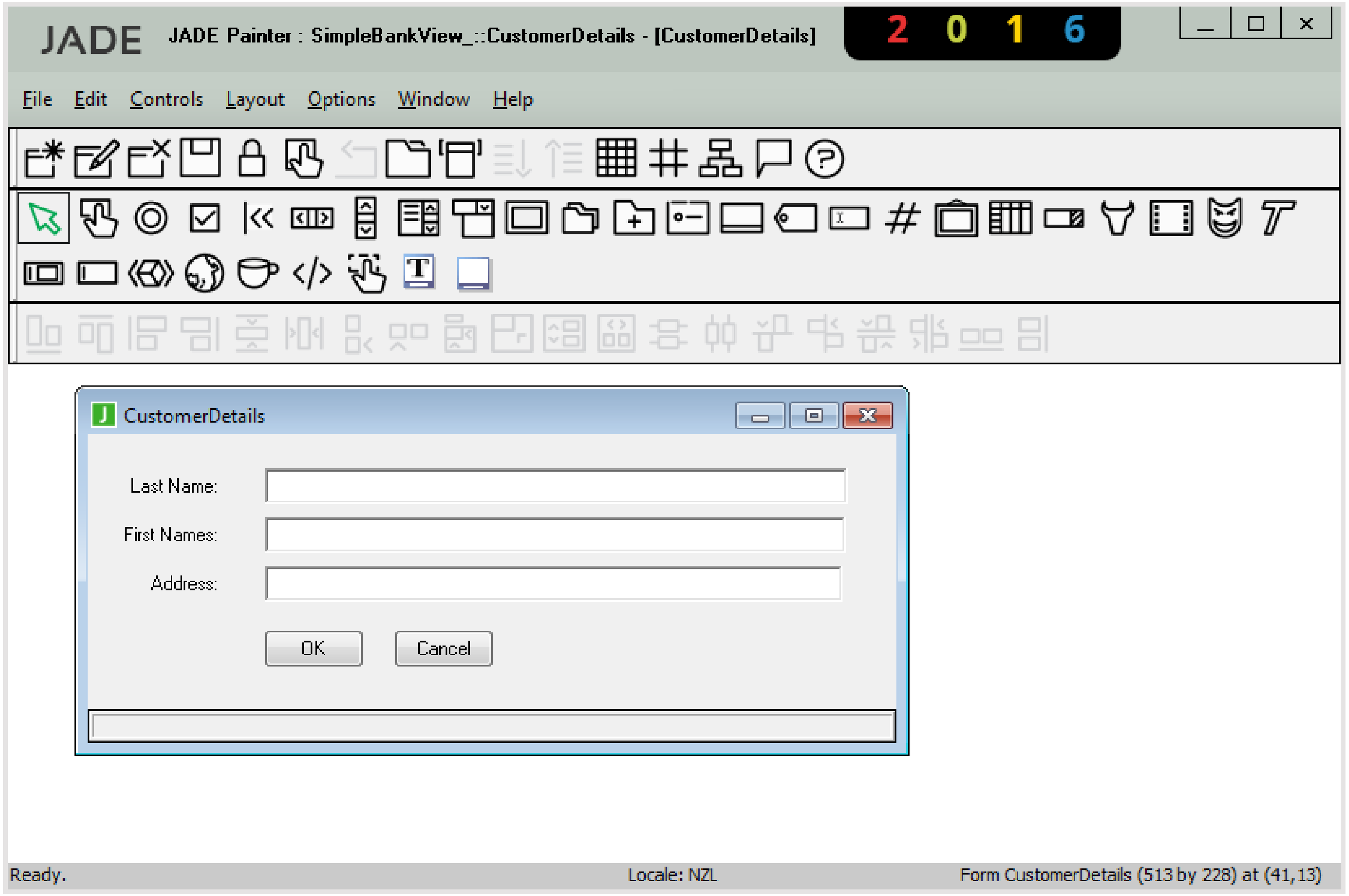
How do we make our UI *usable*?

1. Learnability - ???
2. Memorability - ???
3. Efficiency - ???
4. Error Recovery - ???
5. Satisfaction - ???

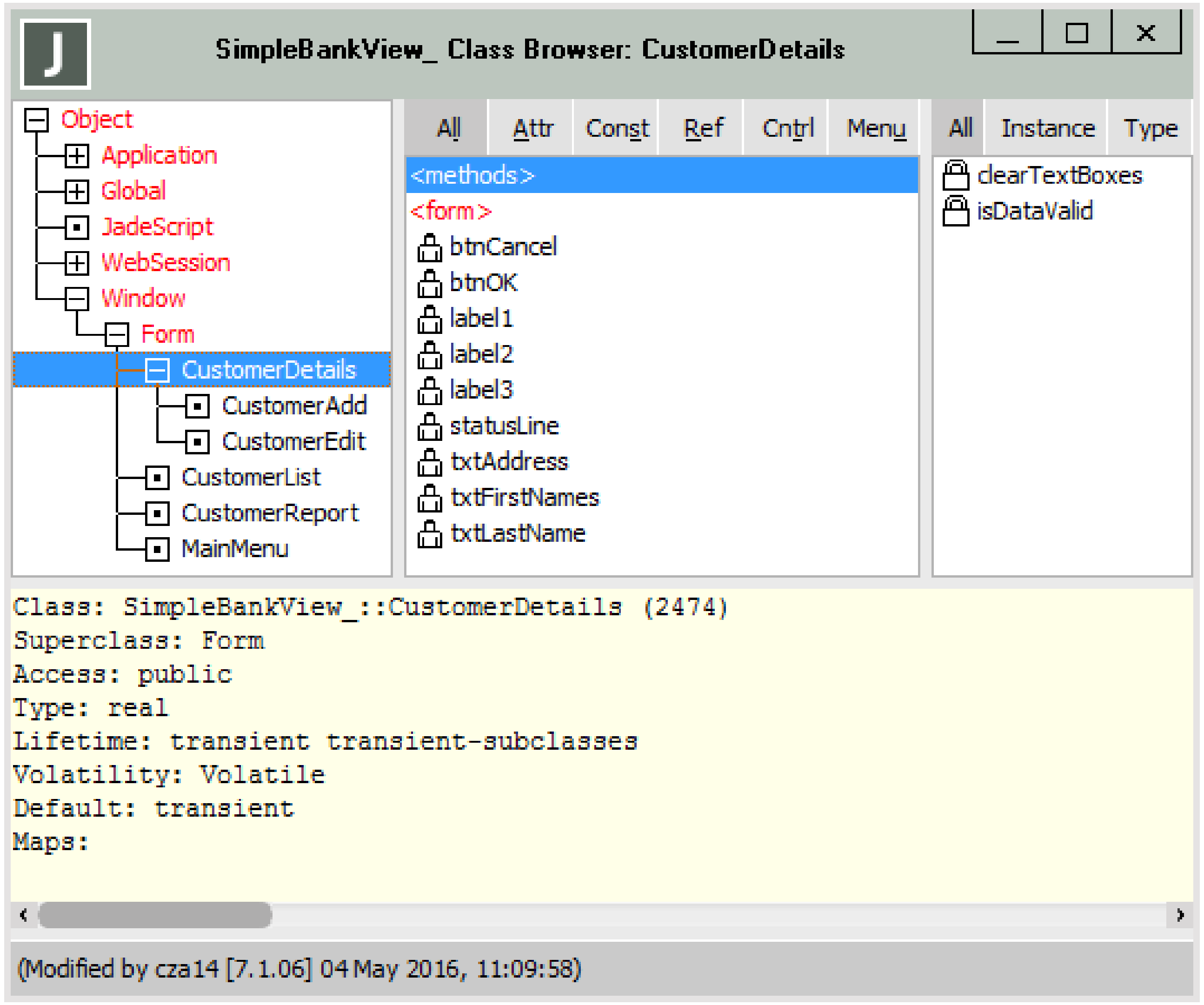
How do we make our UI *usable*?

1. Learnability - How easy is it for newbies?
2. Memorability - What about 2nd time?
3. Efficiency - How fast can I go?
4. Error Recovery - What if I mess up?
5. Satisfaction - Is it pleasant to use?

GUI Design+Development in JADE


JADE
JADE Painter : SimpleBankView_::CustomerDetails - [CustomerDetails]
2 0 1 6
File Edit Controls Layout Options Window Help
CustomerDetails
Last Name:
First Names:
Address:
OK
Cancel
Ready.
Locale: NZL
Form CustomerDetails (513 by 228) at (41,13)

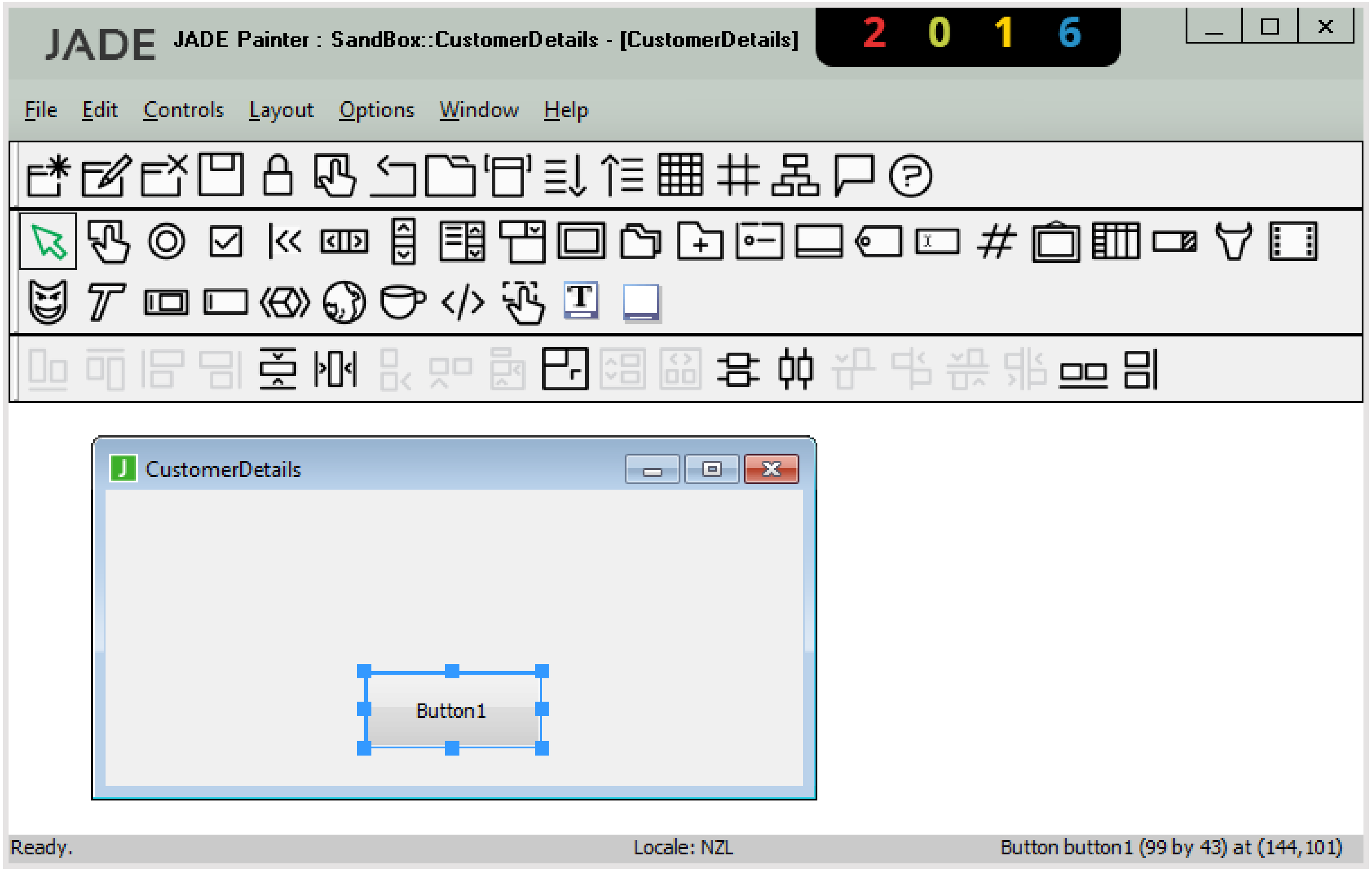
Seamlessly integrates with JADE code



```
Class: SimpleBankView_::CustomerDetails (2474)
Superclass: Form
Access: public
Type: real
Lifetime: transient transient-subclasses
Volatility: Volatile
Default: transient
Maps:
```

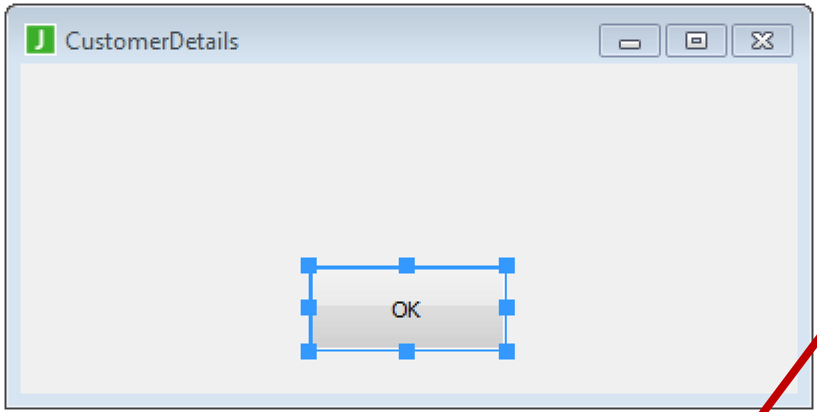
(Modified by cza14 [7.1.06] 04 May 2016, 11:09:58)

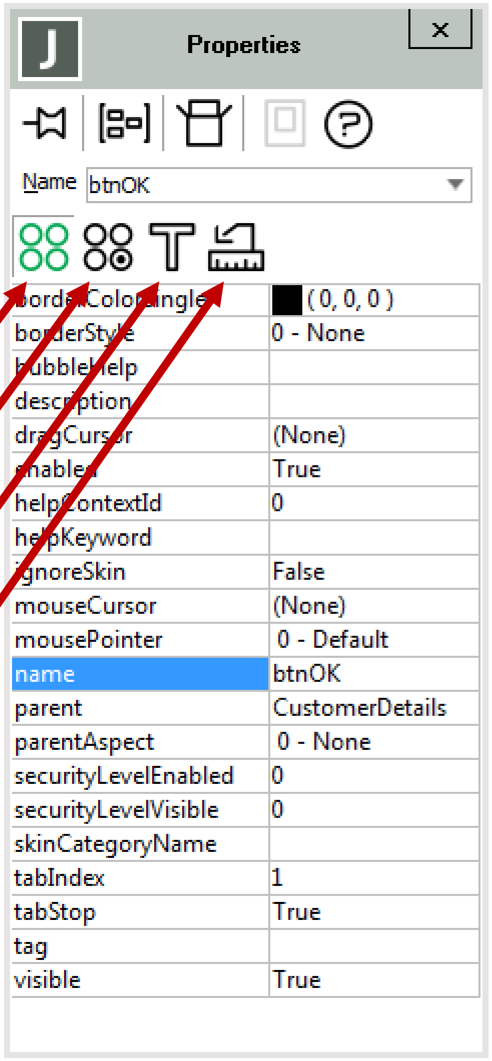
Point and Click Interface Component Placement


JADE
JADE Painter : SandBox::CustomerDetails - [CustomerDetails]
2 0 1 6
File Edit Controls Layout Options Window Help
CustomerDetails
Button1

Ready.   Locale: NZL   Button button1 (99 by 43) at (144,101)

Properties Dialog (E.g., for a Button)





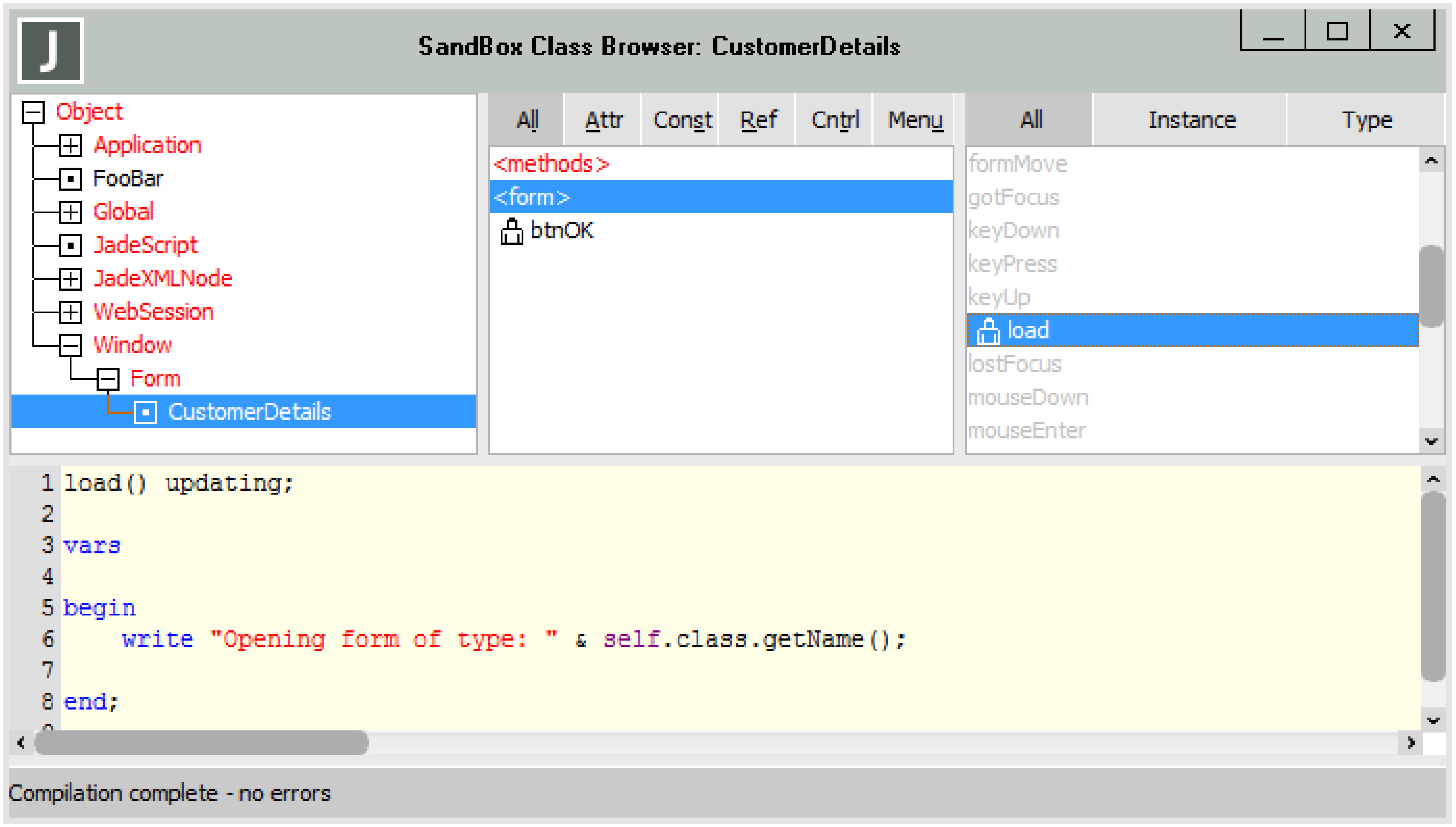
Name btnOK

borderColorSingle	■ (0, 0, 0)
borderStyle	0 - None
bubbleHelp	
description	
dragCursor	(None)
enabled	True
helpContextId	0
helpKeyword	
ignoreSkin	False
mouseCursor	(None)
mousePointer	0 - Default
name	btnOK
parent	CustomerDetails
parentAspect	0 - None
securityLevelEnabled	0
securityLevelVisible	0
skinCategoryName	
tabIndex	1
tabStop	True
tag	
visible	True

Properties groups/tabs:

- Common
- Specific
- Font and colour
- Size and position

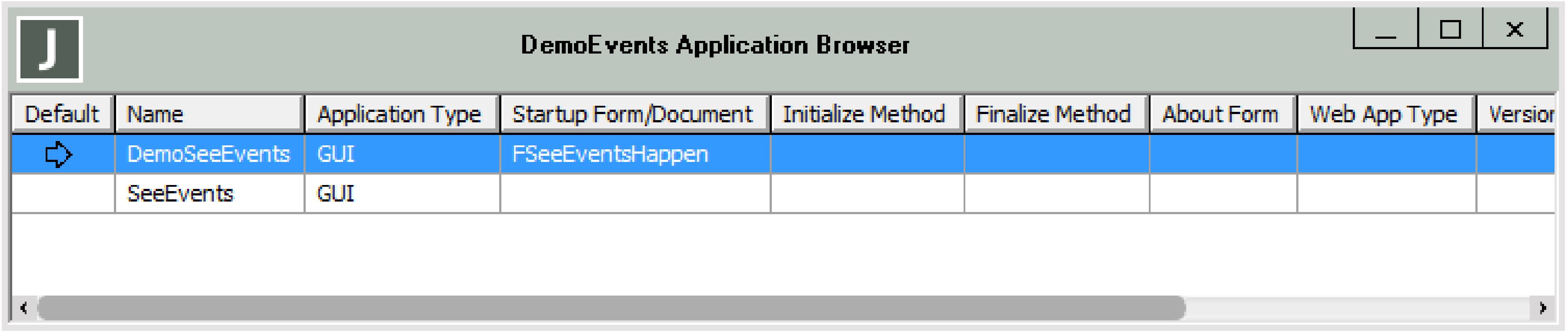
Inherited Method Overriding



```
load() updating;

vars

begin
    write "Opening form of type: " & self.class.getName();

end;
```

Compilation complete - no errors

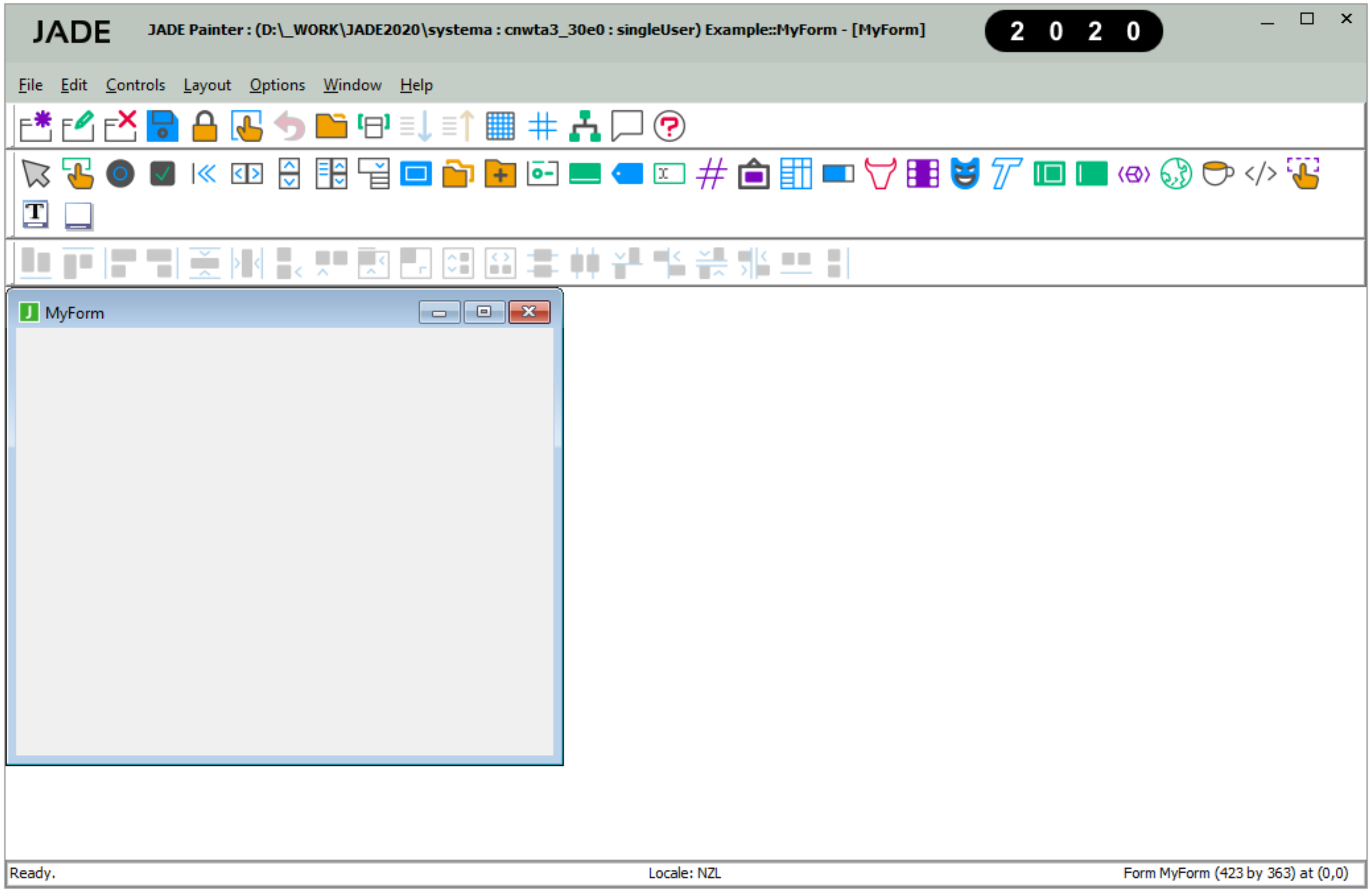
JADE IDE: Applications Browser


DemoEvents Application Browser

Default	Name	Application Type	Startup Form/Document	Initialize Method	Finalize Method	About Form	Web App Type	Versior
⇨	DemoSeeEvents	GUI	FSeeEventsHappen					
	SeeEvents	GUI						

- Application Browser brings together all applications defined in a schema
- GUI and non-GUI applications options can be defined in the same schema

DEMO: Creating a JADE Form


JADE
JADE Painter : (D:_WORK\JADE2020\systema : cnwta3_30e0 : singleUser) Example::MyForm - [MyForm]
2 0 2 0
File Edit Controls Layout Options Window Help
MyForm
Ready.
Locale: NZL
Form MyForm (423 by 363) at (0,0)

Action Points, Readings for Next Week

- Make a start on the assignment!
  - Consider what extra information you might need…
  - Create the classes/references as per the class diagram
  - Start thinking about what the user interfaces will look like
  - Make a use case diagram?