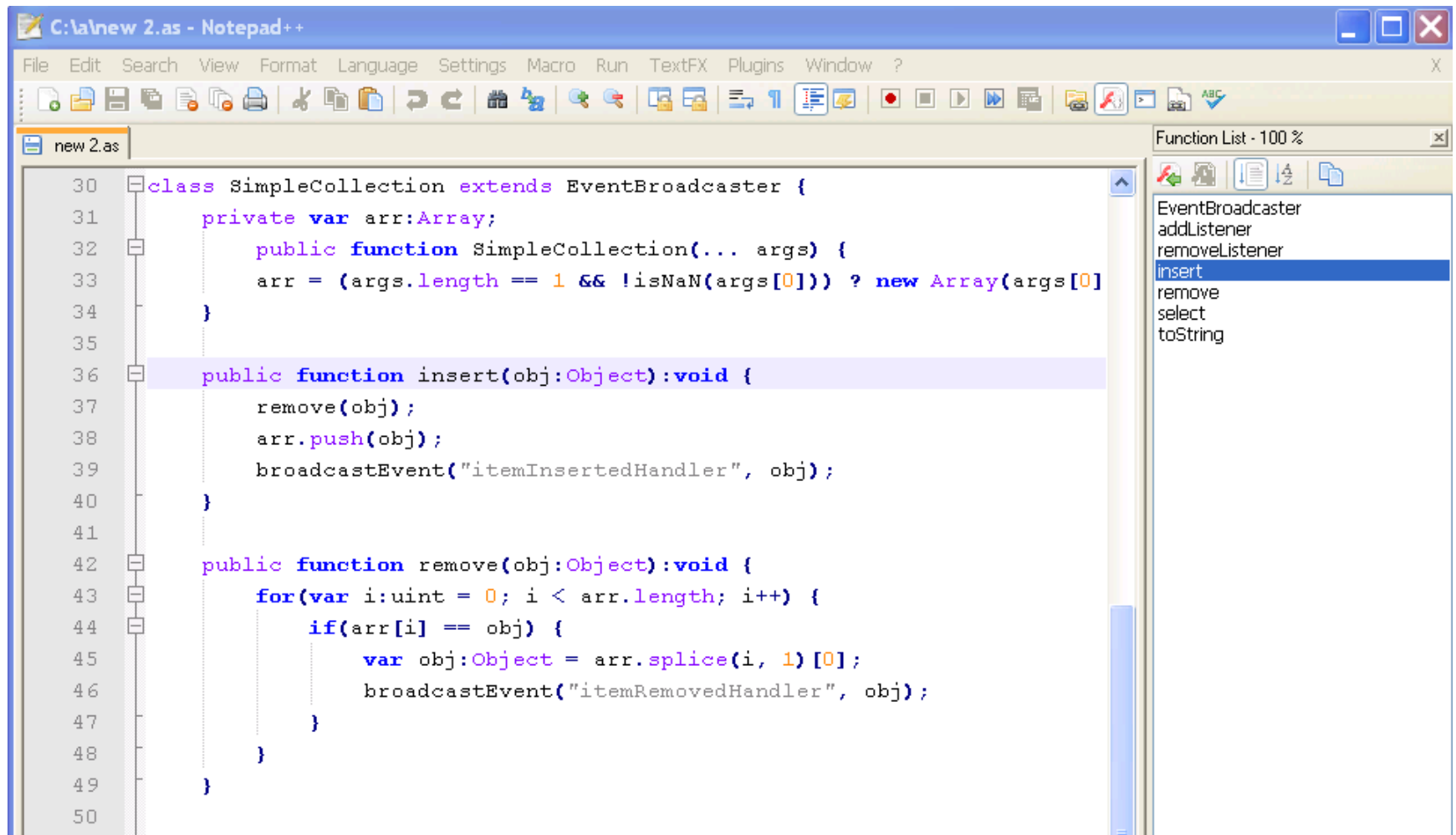


C r e a t i n g f u n c t i o n l i s t f o r u s e r
d e f i n e d l a n g u a g e i n N o t e p a d + +

Flash ActionScript

- Following slides illustrate how to achieve in 5 steps result

Flash ActionScript



The image shows a Notepad++ window titled "C:\a\new 2.as - Notepad++". The main editor area contains the following ActionScript code:

```
30 class SimpleCollection extends EventBroadcaster {
31     private var arr:Array;
32     public function SimpleCollection(... args) {
33         arr = (args.length == 1 && !isNaN(args[0])) ? new Array(args[0]
34     }
35
36     public function insert(obj:Object):void {
37         remove(obj);
38         arr.push(obj);
39         broadcastEvent("itemInsertedHandler", obj);
40     }
41
42     public function remove(obj:Object):void {
43         for(var i:uint = 0; i < arr.length; i++) {
44             if(arr[i] == obj) {
45                 var obj:Object = arr.splice(i, 1)[0];
46                 broadcastEvent("itemRemovedHandler", obj);
47             }
48         }
49     }
50 }
```

The "insert" function is currently selected. To the right, a "Function List" panel is open, showing a list of methods: EventBroadcaster, addListener, removeListener, insert (highlighted), remove, select, and toString.

Flash ActionScript

The image shows a Notepad++ window titled "C:\a\new 2.as - Notepad++" with a menu bar (File, Edit, Search, View, Format, Language, Settings, Macro, Run, TextFX, Plugins, Window) and a toolbar. The main editor displays the following ActionScript code:

```
30 class SimpleCollection extends EventBroadcaster {
31     private var arr:Array;
32     public function SimpleCollection(... args) {
33         arr = (args.length == 1 && !isNaN(args[0])) ? new Array(args[0])
34     }
35
36     public function insert(obj:Object):void {
37         remove(obj);
38         arr.push(obj);
39         broadcastEvent("itemInsertedHandler", obj);
40     }
41
42     public function remove(obj:Object):void {
43         for(var i:uint = 0; i < arr.length; i++) {
44             if(arr[i] == obj) {
45                 var obj:Object = arr.splice(i, 1)[0];
46                 broadcastEvent("itemRemovedHandler", obj);
47             }
48         }
49     }
50 }
```

A red box highlights the `insert` function signature on line 36. A red arrow points from this box to the `insert` entry in the Function List on the right. The Function List, titled "Function List - 100 %", contains the following entries:

- EventBroadcaster
- addListener
- removeListener
- insert**
- remove
- select
- toString

Flash ActionScript

The image shows a Notepad++ window titled "C:\a\new 2.as - Notepad++" with a menu bar (File, Edit, Search, View, Format, Language, Settings, Macro, Run, TextFX, Plugins, Window) and a toolbar. The main editor displays the following ActionScript code:

```
30 class SimpleCollection extends EventBroadcaster {
31     private var arr:Array;
32     public function SimpleCollection(... args) {
33         arr = (args.length == 1 && !isNaN(args[0])) ? new Array(args[0]
34     }
35
36     public function insert(obj:Object):void {
37         remove(obj);
38         arr.push(obj);
39         broadcastEvent("itemInsertedHandler", obj);
40     }
41
42     public function remove(obj:Object):void {
43         for(var i:uint = 0; i < arr.length; i++) {
44             if(arr[i] == obj) {
45                 var obj:Object = arr.splice(i, 1)[0];
46                 broadcastEvent("itemRemovedHandler", obj);
47             }
48         }
49     }
50 }
```

On the right side, a "Function List - 100%" panel is visible, listing the following functions: EventBroadcaster, addListener, removeListener, insert, remove, select, and toString. The "remove" function is highlighted in blue, and a red box is drawn around it. A red arrow points from the "remove" function name in the code (line 42) to the "remove" function in the Function List.

Flash ActionScript

1. Define predecessor of function name, e.g. `function`

The image shows a Notepad++ window with the file name "C:\new 2.as". The code in the editor is as follows:

```
30 class SimpleCollection extends Event
31     private var arr:Array;
32     public function SimpleCollec
33         arr = (args.length == 1 && !
34     }
35
36     public function Insert(obj:Object
37         remove(obj);
38         arr.push(obj);
39         broadcastEvent("itemInserted
40     }
41
42     public function remove(obj:Object
43         for(var i:uint = 0; i < arr.
44             if(arr[i] == obj) {
45                 var obj:Object = arr
46                 broadcastEvent("item
47             }
48         }
49     }
50
```

The word "function" in line 36 is circled in green. A green arrow points from this "function" to the "Function Begin" field in the "Function List Parsing Rules" dialog box, which contains the text "function[\t]+". This field is also circled in green.

The "Function List Parsing Rules" dialog box is open over the code. It has a "Language" dropdown set to "User Defined" and a "Match Case" checkbox that is unchecked. There are buttons for "Remove rule" and "Help".

Under the "Section Rules" section, there is a list of rules. Rule 1 is selected, and there are "Add Rule" and "Delete Rule" buttons. The fields for rule 1 are:

- Function Begin: function[\t]+
- Function List Name: [a-z_ \t]+
- Function End: \{
- Separator Between: ;
- Body Begin: \{
- Body End: \}

Under the "Comment Rules" section, there are "Add Rule" and "Delete Rule" buttons. The fields for comment rules are:

- Multiline Begin / Singleline: (empty)
- Multiline End: (empty)

Under the "Keyword forwarding" section, there are two empty text boxes:

- Function End to Body Begin: (empty)
- Body Begin to Body End: (empty)

Flash ActionScript

2. Set function name regular expression, e.g. `[a-z_ \t]+`

The image shows a Notepad++ window with the file name "C:\new 2.as". The code in the editor is as follows:

```
30 class SimpleCollection extends Event
31     private var arr:Array;
32     public function SimpleCollec
33         arr = (args.length == 1 && !
34     }
35
36     public function insert(obj:Object
37         remove(obj);
38         arr.push(obj);
39         broadcastEvent("itemInserted
40     }
41
42     public function remove(obj:Object
43         for(var i:uint = 0; i < arr.
44             if(arr[i] == obj) {
45                 var obj:Object = arr
46                 broadcastEvent("item
47             }
48         }
49     }
50
```

The "Function List Parsing Rules" dialog box is open, showing the following settings:

- Language: User Defined
- Match Case:
- Section Rules: 1 (Add Rule, Delete Rule)
- Function Begin: `function[\t]+`
- Function List Name: `[a-z_ \t]+` (highlighted with a green circle)
- Function End: `\{`
- Seperator Between: `;`
- Body Begin: `\{`
- Body End: `\}`
- Comment Rules: Add Rule, Delete Rule
- Multiline Begin / Singleline:
- Multiline End:
- Keyword forwarding: Function End to Body Begin, Body Begin to Body End

Flash ActionScript

3. Define successor of function name, e.g. (

The image shows a Notepad++ window with the file name "C:\new 2.as". The code in the editor is as follows:

```
30 class SimpleCollection extends Event
31     private var arr:Array;
32     public function SimpleCollec
33         arr = (args.length == 1 && !
34     }
35
36     public function insert(obj:Object
37         remove(obj);
38         arr.push(obj);
39         broadcastEvent("itemInserted
40     }
41
42     public function remove(obj:Object
43         for(var i:uint = 0; i < arr.
44             if(arr[i] == obj) {
45                 var obj:Object = arr
46                 broadcastEvent("item
47             }
48         }
49     }
50
```

The "Function List Parsing Rules" dialog box is open, showing the following configuration:

- Language: User Defined
- Match Case:
- Section Rules:
 - 1 (selected)
 - Add Rule
 - Delete Rule
 - Function Begin: function[\t]+
 - Function List Name: [a-z_ \t]+
 - Function End: \{ (circled in green)
 - Seperator Between: ;
 - Body Begin: \{
 - Body End: \}
- Comment Rules:
 - Add Rule
 - Delete Rule
 - Multiline Begin / Singleline:
 - Multiline End:
- Keyword forwarding:
 - Function End to Body Begin:
 - Body Begin to Body End:

A green circle highlights the backslash character in the "Function End" field, with an arrow pointing to the corresponding closing curly brace in the code editor.

Flash ActionScript

4. Set function body **begin** if exists any, e.g. {

The image shows a Notepad++ window with the file 'new 2.as' open. The code is as follows:

```
30 class SimpleCollection extends Event
31     private var arr:Array;
32     public function SimpleCollec
33         arr = (args.length == 1 && !
34     }
35
36     public function insert(obj:Object
37     {
38         remove(obj);
39         arr.push(obj);
40         broadcastEvent("itemInserted
41     }
42
43     public function remove(obj:Object
44     for(var i:uint = 0; i < arr.
45         if(arr[i] == obj) {
46             var obj:Object = arr
47             broadcastEvent("item
48         }
49     }
50
```

The opening curly brace of the `insert` function on line 36 is circled in green. A green arrow points from this brace to the 'Body Begin' field in the 'Function List Parsing Rules' dialog box, which also contains a green circle around the backslash character `\{`.

The 'Function List Parsing Rules' dialog box is open, showing the following settings:

- Language: User Defined
- Match Case:
- Section Rules:
 - 1 (selected)
 - Add Rule
 - Delete Rule
- Function Begin: `function[\t]+`
- Function List Name: `[a-z_ \t]+`
- Function End: `\{`
- Seperator Between: `;`
- Body Begin: `\{` (circled in green)
- Body End: `\}`
- Comment Rules:
 - Add Rule
 - Delete Rule
- Multiline Begin / Singleline:
- Multiline End:
- Keyword forwarding:
 - Function End to Body Begin:
 - Body Begin to Body End:

Flash ActionScript

5. Set function body end if exists any, e.g. }

The image shows a Notepad++ window with the file name 'C:\new 2.as'. The code in the editor is as follows:

```
30 class SimpleCollection extends Event
31     private var arr:Array;
32     public function SimpleCollection()
33     {
34     }
35
36     public function insert(obj:Object)
37     {
38         remove(obj);
39         arr.push(obj);
40         broadcastEvent("itemInserted");
41     }
42
43     public function remove(obj:Object)
44     {
45         for(var i:uint = 0; i < arr.length; i++)
46         {
47             if(arr[i] == obj)
48             {
49                 arr.splice(i, 1);
50                 broadcastEvent("itemRemoved");
51             }
52         }
53     }
54 }
```

The closing brace of the `insert` function on line 41 is circled in green. A green arrow points from this brace to the 'Body End' field in the 'Function List Parsing Rules' dialog box, which also contains a green circle. The dialog box is titled 'Function List Parsing Rules' and has the following settings:

- Language: User Defined
- Match Case:
- Section Rules:
 - 1 (selected)
 - Add Rule
 - Delete Rule
- Function Begin: `function[\t]+`
- Function List Name: `[a-z_ \t]+`
- Function End: `\{`
- Separator Between: `;`
- Body Begin: `\{`
- Body End: `\}` (circled in green)
- Comment Rules:
 - Add Rule
 - Delete Rule
- Multiline Begin / Singleline:
- Multiline End:
- Keyword forwarding:
 - Function End to Body Begin:
 - Body Begin to Body End: