DataBinding - поглед в генерирания код

obecto
passionate about software & people

Before pre-processing

Product.as

```
package vo
{
	[Bindable]
	public class Product
	{
		public var name:String;
		public var price:Number;
	}
}
```

demo.mxml

```
<?xml version="1.0" encoding="utf-8"?>
<s:Application
	xmlns:vo=”vo.*”>

	<fx:Declarations>
		<vo:Product id="product" name="Apricots" price=”5.5”/>
	</fx:Declarations>

	<s:Label id="productLabel" text="{product.name}"/>

</s:Application>
```

obecto
passionate about software & people

After pre-processing

```
_Product-binding-generated.as

demo-generated.as

_demoWatcherSetupUtil.as
```


_Product-binding-generated.as

```
class BindableProperty implements flash.events.IEventDispatcher
{
    [Bindable(event="propertyChange")]
    public function get name():String
    {
        return this._1491817446name;
    }

    public function set name(value:String):void
    {
        var oldValue:Object = this._1491817446name;
        if (oldValue !== value)
        {
            this._1491817446name = value;
            this.dispatchEvent(mx.events.PropertyChangeEvent.createUpdateEvent(
                this, "name", oldValue, value));
        }
    }

    private var _bindingEventDispatcher:flash.events.EventDispatcher =
        new flash.events.EventDispatcher(flash.events.IEventDispatcher(this));

    //...
}
```

obecto
passionate about software & people

demo-generated.as

```
public function demo()
{
	super();
	mx_internal::_document = this;

	var bindings:Array = _demo_bindingsSetup();
	var watchers:Array = [];

	var target:Object = this;

	if (_watcherSetupUtil == null)
	{
		var watcherSetupUtilClass:Object = getDefinitionByName("_demoWatcherSetupUtil");
		watcherSetupUtilClass["init"](null);
	}

	_watcherSetupUtil.setup(this,
		function(propertyName:String):* { return target[propertyName]; },
		function(propertyName:String):* { return demo[propertyName]; },
			bindings,
			watchers);

	mx_internal::_bindings = mx_internal::_bindings.concat(bindings);
	mx_internal::_watchers = mx_internal::_watchers.concat(watchers);

	_demo_Product1_i();

	for (var i:uint = 0; i < bindings.length; i++)
	{
		Binding(bindings[i]).execute();
	}
}
```

demo-generated.as

```
private function _demo_bindingsSetup():Array
{
    var result:Array = [];

    result[0] = new mx.binding.Binding(this,
        function():String {
            var result:* = (product.name);
            return (result == undefined ? null : String(result));
        },
        null,
        "productLabel.text"
    );

    return result;
}
```

obecto
passionate about software & people

demo-generated.as

```
private function _demo_bindingsSetup():Array
{
    var result:Array = [];

    result[0] = new mx.binding.Binding(this,
        function():String {
            var result:* = (product.name);
            return (result == undefined ? null : String(result));
        },
        null,
        "productLabel.text"
    );

    return result;
}
```

mx.binding.Binding

- това е обекта отговорен да изпълнява *binding*-а
- *srcFunc* е функцията, която връща стойността, която трябва да се присвои
- *destFunc* е функцията, която извършва присвояването
- *binding* обектите не се задействат автономно - някой ги режисира (кой е този някой, ще видим след малко)

obecto
passionate about software & people

demo-generated.as

```
public function demo()
{
super();
mx_internal::_document = this;

	var bindings:Array = _demo_bindingsSetup();
	var watchers:Array = [];

	var target:Object = this;

	if (_watcherSetupUtil == null)
	{
	var watcherSetupUtilClass:Object = getDefinitionByName("_demoWatcherSetupUtil");
		watcherSetupUtilClass["init"](null);
	}

	_watcherSetupUtil.setup(this,
		function(propertyName:String):* { return target[propertyName]; },
		function(propertyName:String):* { return demo[propertyName]; },
			bindings,
		watchers);

	mx_internal::_bindings = mx_internal::_bindings.concat(bindings);
	mx_internal::_watchers = mx_internal::_watchers.concat(watchers);

	_demo_Product1_i();

	for (var i:uint = 0; i < bindings.length; i++)
	{
		Binding(bindings[i]).execute();
	}
}
```

_demoWatcherSetupUtil.as

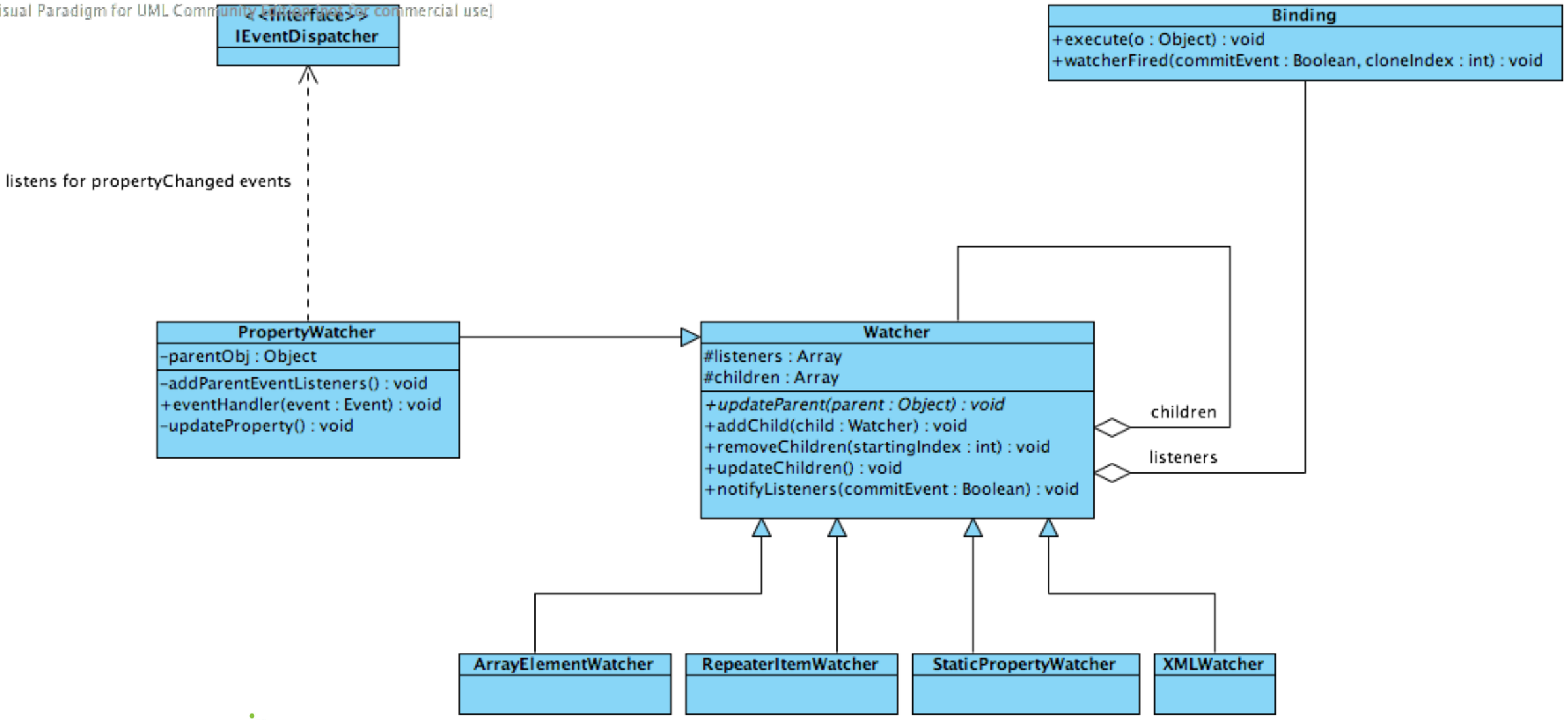
```
public function setup(target:Object,
            propertyGetter:Function,
            staticPropertyGetter:Function,
            bindings:Array,
            watchers:Array):void
{
        import mx.core.IFlexModuleFactory;
        watchers[0] = new mx.binding.PropertyWatcher(
            "product",
            {
                propertyChange: true
            },
            [bindings[0]],
            propertyGetter);

        watchers[1] = new mx.binding.PropertyWatcher(
            "name",
            {
                  propertyChange: true
            },
            [bindings[0]],
            null);

        watchers[0].updateParent(target);
        watchers[0].addChild(watchers[1]);
}
```

***PropertyWatcher* е отговорен да забележи промени в стойностите и да уведоми *Binding*-обектите, които трябва да се изпълнят**

obecto
passionate about software & people

Да съединим точките...

obecto
passionate about software & people



Обобщение

- какво се генерира на мястото на *[Bindable]*
- *mx.binding.Binding* обектите
- *PropertyWatch* обектите
- UML диаграма на целия механизъм