JavaOne℠

Sun's Worldwide Java Developer Conference

Java Beans Component APIs for Java

Graham Hamilton
JavaSoft

Java™ Beans

- Java beans is a set of component APIs
  - Allowing pluggable software components
  - In an open, portable, platform-neutral way
- Java beans covers a range of components
  - From simple lightweight widgets
  - To powerful components
  - To full scale applications

Overview

- Why Java needs component APIs
- What Java Beans supports
- Implementing Java Beans in different environments

(Note: Java Beans is the current code name)

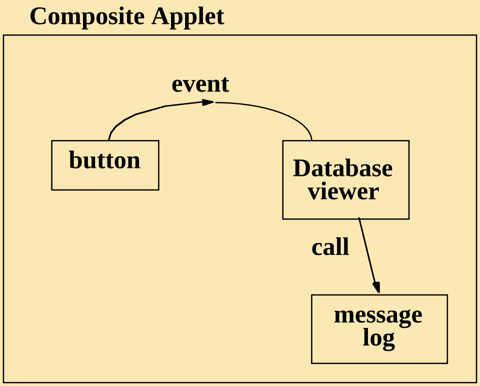
Why a Component Model?

- In the past ISVs wrote entire applications
  - But this means they duplicate a lot of code
  - And their applications can’t plug together
- Customers want to plug together components
  - And compose them into applications
  - And use some components to control others

Java and Components

- Applets are a good, simple component model
  - They allow simple static integration
- We'd like to enable richer, dynamic integration:
  - So that components can interact
    - By firing and catching events
    - By calling methods on one another
  - So that components can merge GUI features
- Without giving up platform portability

Java Components



Java Beans as Components

- Can raise and catch typed events
- Can support persistent properties
- Can be stored as part of a parent
- Can act as servers for other components
- Can merge their GUI with their parent
  - e.g. they may merge MenuBars with parent
- Can provide “component editors” for builders
  - To allow users to customize component behavior

Java Beans as Containers

- Beans can be containers for other beans
- This allows hierarchical components
- And allows Java compound documents

Keeping it Simple

- Java beans can be very lightweight
  - All Java AWT components will be beans
  - All AWT containers will be bean containers
- But some beans will be much larger
  - Spreadsheets embedded in word processors
- The Java beans APIs are focused on making the simple cases easy
  - And the hard cases doable

Keeping it Simple

- We don’t want to complicate simple AWT components
- So runtimes provide default behavior
  - But allow an object to override this behavior
  - By testing for “instanceof java.beans.XXX”
  - This avoids having heavyweight base classes

Related new APIs

- Improved desktop integration from AWT
  - Drag-and-drop
  - Cut-and-paste
- Java RMI: remote method invocation
- Industry standard CORBA IDL for remote object access
- Automatic object serialization

Portability

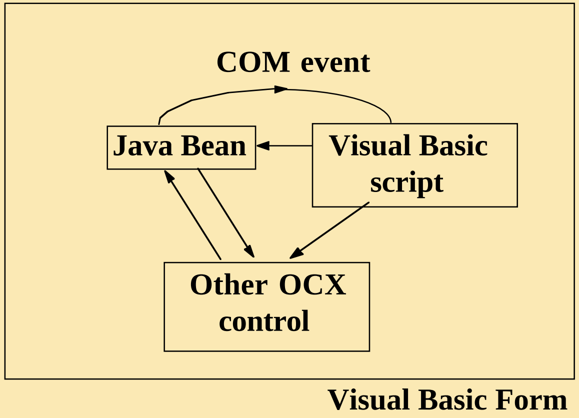
- Java beans, a platform neutral Java API
- A single Java bean can integrate in a high quality way into a variety of containers:
  - HotJava™ and other Java containers
  - MS containers (e.g. Word, Visual Basic, Explorer)
  - Netscape (using JavaScript™&LiveConnect)
  - Opendoc containers
  - And will also run outside of any containers

"Write once, run anywhere" at a new level

COM Implementation

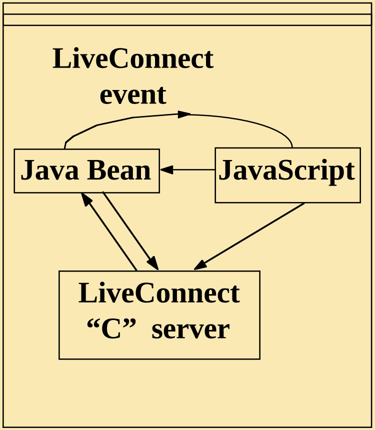
- Libraries will bridge Java bean API to COM
- Using standard COM APIs:
  - OLE automation, OLE documents, ActiveX
- Java bean events get mapped to COM events
- There will be a Java bean OCX
- Java beans will appear as “first class” COM and ActiveX components
  - While still being completely portable

Beans in Visual Basic



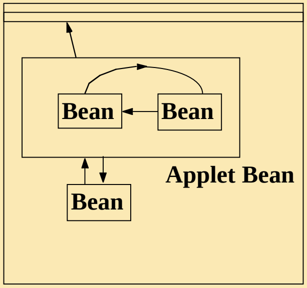
Netscape Navigator

- Java bean libraries will bridge Java bean APIs to JavaScript and LiveConnect



HotJava

- HotJava will be extended to support the full Java bean container API
- This will provide a much higher degree of integration of applets into HotJava



Hot Java

Conclusion

- Java Beans is an open Java component model
- Beans are genuinely platform portable
- They integrate into platform component model
- Java Beans will allow you to write portable Java components with first-class platform integration!

JavaOne℠

Sun's Worldwide Java Developer Conference