Exploring Web Dynpro Java: Goals, Concepts, Benefits

SAP NetWeaver CE 7.11 EhP1

Bertram Ganz, SAP NetWeaver UI Foundation OPS

November 2008
Disclaimer

This presentation outlines our general product direction and should not be relied on in making a purchase decision. This presentation is not subject to your license agreement or any other agreement with SAP. SAP has no obligation to pursue any course of business outlined in this presentation or to develop or release any functionality mentioned in this presentation. This presentation and SAP's strategy and possible future developments are subject to change and may be changed by SAP at any time for any reason without notice. This document is provided without a warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement. SAP assumes no responsibility for errors or omissions in this document, except if such damages were caused by SAP intentionally or grossly negligent.
**Presentation Abstract**

**Exploring Web Dynpro Java: Goals, Concepts, Benefits**

<table>
<thead>
<tr>
<th>Release</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAP NetWeaver 7.0 and CE 7.1</td>
<td>In this presentation you will explore the real goals, concepts and benefits of the Web Dynpro Java UI framework for building business web applications. We highlight particular features of Web Dynpro like its client-agnostic user interface concept to run the same application on different client-devices, its rich UI element library with in-built Ajax functions, its value help, message and personalization services, its model-based development environment with maximized code-generation, its component model, its integration into the SAP NetWeaver Portal and its model interface to consistently integrate EJBs, Web Services or RFC modules. We illustrate how these functions yield benefits in respect of UI consistency, client-device-independence, accessibility, component reuse as well as reduced total costs of development and maintenance. We further clarify the key differentiators of Web Dynpro Java compared to other Web frameworks like JSF and demonstrate its openness to rich client UI technologies like Adobe Flash by using the new Web Dynpro Flash Islands introduced in SAP NetWeaver CE 7.1. We finally conclude this session with a short comparison to the other Web Dynpro flavors Web Dynpro for Visual Composer and Web Dynpro for ABAP.</td>
</tr>
</tbody>
</table>

© SAP 2008 / SAP NetWeaver CE 7.1 EhP1 / Exploring Web Dynpro Java: Goals, Concepts, Benefits / Page 3
1. Download related Exercises from SCN here ...

2. Download related Software Component Archive from SCN here and import comprised Web Dynpro Java DCs into your SAP NetWeaver Developer Studio

3. Process 3 Exercises ...
   1. Exploring a Very Simple Web Dynpro Java Application
   2. Using the AnalyticsChart UI Element
   3. Using an EJB Model in a Web Dynpro Java Application
Exploring Web Dynpro Java: Goals, Concepts, Benefits
Agenda

Exploring Web Dynpro Java: Goals, Concepts, Benefits

1. Web Dynpro Basics at a Glance
2. Programming Model
3. User Interface
4. Component Model
5. Backend Connectivity
6. Web Dynpro Flavors and Differentiators
1. Web Dynpro Goals and Concepts at a Glance
2. Programming Model
3. User Interface
4. Component Model
5. Backend Connectivity
6. Web Dynpro Flavors and Differentiators
Web Dynpro Java is a proprietary SAP NetWeaver UI technology. Although it applies many common architectural and conceptual paradigms of other serverside Web UI frameworks like Struts or JSF it defines its own programming and component model.

**Web Dynpro Tools:** the Web Dynpro design-time environment inside SAP NetWeaver Developer Studio

**Web Dynpro Java Downward Compatibility Contracts**

1. **Intra-Release Compatibility Contract:** *Backward Binary Compatible*

Run Web Dynpro applications developed on the same release (like SAP NetWeaver CE 7.1) on an earlier Service Pack stack (like SP5) on a higher stack just with re-deployment but without re-compilation

2. **Inter-Release Compatibility Contract:** *Backward Source Compatible*

Run Web Dynpro applications developed in an earlier release (like SAP NetWeaver 7.0) on a higher release (like SAP NetWeaver CE 7.1) with re-generation, re-compilation and re-deployment but without source code modification.

This means you can still run existing Web Dynpro applications developed in SAP NetWeaver 04 and SAP NetWeaver 7.0 on a SAP NetWeaver CE 7.1 AS Java runtime environment.
Web Dynpro Java Product History
From 2003 to Now

CCTS Datatypes, Calendar UI elements, context menus, streamlined component model

 Implicit personalization, suspend/resume application interoperability, table enrichments, new UI elements, Adaptive WS Model

SAP NetWeaver
04

SAP NetWeaver
7.0

SAP NetWeaver
CE 7.1

SAP NetWeaver
CE 7.11 EhP1

Service Pack Stack 16

Service Pack Stack 22

2003 2006 2007 2008

Web Dynpro Ajax Client, Drag&Drop, EJB 3 Model

Foundation UI technology for SAP Business ByDesign

Reached 1000 direct customers

Foundation UI Technology for SAP applications like ESS/MSS

Implicit personalization, suspend/resume application interoperability, table enrichments, new UI elements, Adaptive WS Model

Foundation UI Technology for WD4VC
The Goals and Concepts of Web Dynpro were set in 2001 when SAP developed a new UI technology for web-based applications.

These aims were regularly, and increasingly implemented and revised in Web Dynpro UI technology, beginning with SAP NetWeaver 2004 and 7.0 through to SAP NetWeaver CE 7.1.
### Web Dynpro Java UI Technology

#### Goals and Concepts at Glance

<table>
<thead>
<tr>
<th>Model-Driven Development</th>
<th>Client Abstraction</th>
</tr>
</thead>
<tbody>
<tr>
<td>MVC Programming Model</td>
<td><em>Client-Device-Independance</em></td>
</tr>
<tr>
<td>Ease of Development based on Declaration and Code-Generation</td>
<td><em>Predefined UI-Element-Library</em></td>
</tr>
<tr>
<td>Web Dynpro Tools</td>
<td><em>Generic UI Services, Valuehelp, I18N, P13N</em>, Accessibility*</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Standard SAP NetWeaver Web-UI-Framework</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Model Abstraction</th>
<th>Component Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Common Model Interface</td>
<td><em>Reuse, Maintainability, Extensibility, Flexibility</em></td>
</tr>
<tr>
<td>Separation of Business Logic and Presentation Logic</td>
<td><em>Loose Component Coupling using Interface Definitions</em></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Component Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAP NetWeaver Web-UI-Framework</td>
</tr>
<tr>
<td>-----------------</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

* I18N = Internationalization  
** P13N=Personalization

### Web Dynpro’s Main Goals

**Deliver an Enterprise Quality Web Development Environment**

- Declarative, model-based UI development: minimize coding, maximize design
- integration into software logistics
- support reuse of components
- separate UI logic (layout) and business logic (RFC, Web Service, Enterprise Service, EJB)

**Achieve Independence...**

- from platform/language: Web Dynpro for Java / Web Dynpro for ABAP
- from UI client technology: Browser, Rich Client, Mobile Device

**Improve User Experience through a "High Fidelity Web UI"**

- Standardized UI elements for a common look and feel
- Flicker-free screen updates w/o page reloads
- personalization for end users
- I18N
- Valuehelp
**Agenda**

*Exploring Web Dynpro Java: Goals, Concepts, Benefits*

1. Web Dynpro Basics at a Glance
2. **Programming Model**
3. User Interface
4. Component Model
5. Backend Connectivity
6. Web Dynpro Flavors and Differentiators
Model View Controller

MVC is for ‘Model View Controller.’ A fundamental aspect of this is the division of a Web application into various areas or components according to three aspects: Model (business logic), View (user interface), and Controller (presentation and application logic).

The **Model-View-Controller is a well-known architectural pattern in computer software development** first described by Trygve Reenskaug in his article: “Thing-Model-View-Editor—An Example from a Planning System.” Technical note, Xerox PARC, (May 1979). Meanwhile, this pattern is the de facto architectural standard adopted by many existing Web programming frameworks in different flavors. One flavor is the Web Dynpro Java programming model based on components, views, controllers, and models.
The 80/20 Model/Code Principle of Web Dynpro Java Programming

Answer

Web Dynpro Java follows the **80/20 Model/Code principle**.

This means 80% of the repetitive UI coding tasks are fulfilled by **generic** or **generated** code. The generated code is based on declarations made at design time (80% Model).

You only have to implement the remaining 20% of Java code inside specific areas of the generated controller classes.

This 20% is made up of those parts of the application logic which cannot be generated from declarations alone – such as:

- Handling user events
- Triggering navigation
- Executing of business methods
- Dynamic adaptation of the user interface

**Question**

You told us that Web Dynpro Java is **model-driven** and **declaration-centric** on one side. On the other side you pointed out the **minimized coding effort**. What does this exactly mean?

In Web Dynpro, the source code to be implemented manually by the application developer is reduced to the actual application logic, including handling user actions, activating navigation links, or calling business logic. All remaining code elements, such as various controller classes or interfaces, are automatically created by the **Web Dynpro generation framework**. This approach significantly reduces the occurrence of errors and the amount of development effort required for the application.

**Web Dynpro Generation Framework**

Minimizes custom coding efforts by generating entities based on the defined meta-data like controller classes and interfaces, classes contain user coding areas where additional code (e.g. event handlers) can be implemented (custom coding), interfaces for context objects (typed nodes and node elements), interfaces for message texts or model classes
The 80/20 Model/Code Principle of Web Dynpro Java Development in Detail

Model and Declare
- Application creation
- Component creation
- Component usage
- Controller definition
- Event subscription
- Context definition
- Context mapping
- Data binding
- Action binding
- Parameter Mapping
- View design and layout
- Navigation flow definition
- Message pool
- Dictionary types
- Model import
- Context-to-model binding

Apply Dynamic Programming*
- Dynamic view modification
- Dynamic context creation
- Dynamic context mapping
- Dynamic model binding
- Dynamic model instance
- Dynamic event subscription
- Dynamic event content
- Dynamic model composition
- Dynamic component usage

Implement Code
- Controller initialization
- Action event handling
- Navigation logic
- Business method execution
- Context modification
- Context attribute calculation
- Serverside eventing
- Web Dynpro service class invocation

*Only needed for specific use cases

Web Dynpro Tools: To Model and Declare Web Dynpro Java applications the Web Dynpro Tools as part of the SAP NetWeaver Developer Studio IDE are inevitable needed. This means Web Dynpro Java applications cannot be developed using a pure XML editor only.

Metamodel Concept and Declarative Programming
A Web Dynpro application is developed using a declarative programming approach. Within the SAP NetWeaver Development Studio there are special tools that allow you to define the architecture of your application without needing to write any code. This representation is known as a metamodel.

The application metamodel is then used to generate skeleton coding that provides the framework within which you can create your own custom business logic.

Once your application has been built and deployed to the SAP AS Java, it executes within a standard server-side runtime environment known as the Web Dynpro Framework (WDF).

Metamodel Declarations
Guarantees that all applications have a standardized architecture. Good for graphical tool support (Screen Layout and Nesting, Navigation and Error Handling, Data Flow, Componentization etc...)

Custom Coding
Custom business logic implemented within the framework of generated code. Good for data-driven, dynamic apps (Implementation of business rules, Dynamic screen modifications, Access to backend systems (via models), Portal eventing, etc...)
Web Dynpro is SAP’s standard toolset for UI development and execution.
A Simple Web Dynpro Application Explained

How Does It Work?

1. Enter Name
2. Press Button
3. Report Message
All Web Dynpro application development is performed using the Web Dynpro perspective of the NetWeaver Developer Studio.

Web Dynpro Application Entity

This special entity in the Web Dynpro programming model defines which visual interface of a Web Dynpro component the application’s startup phase addresses. You can start the application via a URL in a Web Dynpro client (e.g., in the Web Dynpro HTML client).

Technically, a Web Dynpro application points to a startup navigation plug of a component interface view that the Web Dynpro root component exposes. The Web Dynpro root component is the root of a Web Dynpro component hierarchy.
A Simple Web Dynpro Application Explained
Create Application and Component in IDE (2)

Web Dynpro Component:
- Implements user interface in view layout
- Stores user input in view controller context
- Implements user action handling in view controller
- Defines message text
- Reports message in view controller

Web Dynpro Application:
- Points to the visual interface of our Web Dynpro Sample component
- Can be started via URL in Web Dynpro (Ajax) Client

Web Dynpro Application Entity

This special entity in the Web Dynpro programming model defines which visual interface of a Web Dynpro component the application’s startup phase addresses. You can start the application via a URL in a Web Dynpro client (e.g., in the Web Dynpro HTML client).

Technically, a Web Dynpro application points to a startup navigation plug of a component interface view that the Web Dynpro root component exposes. The Web Dynpro root component is the root of a Web Dynpro component hierarchy.
A Simple Web Dynpro Application Explained
Implement Interface View in Window

Component Interface View
- Exposes the user interface implemented inside a Web Dynpro component to outside users like a Web Dynpro application or another component

Window:
- Implements component interface view
- Embeds views like our SampleView here

Web Dynpro Window
Web Dynpro windows provide the possibility to merge several views to an entire user interface. A window contains the definition of a view composition that is defined by the connection of views or component interface views of embedded components to navigation links. By connecting inbound and outbound plugs at the view level, you define the navigation schema in a window.

Additionally a window and its associated window controller can act as implementers of one or more defined component interface views. Therefore, navigation plugs of type startup, exit, inbound, outbound, suspend and resume defined on window level can be fired and handled in the associated window controller.

Component Interface View = visual component interface

The purpose of a component interface view is to expose the application user interface implemented inside a visual Web Dynpro component via the UI for external use. You can use a visual interface of a Web Dynpro component to build modularized UIs: You split the application user interface into separate Web Dynpro UI components, each implementing a specific part of the application user interface.

A Web Dynpro component implements a defined component interface view with one of its windows, so that the internal window is made visible externally. Another component can then embed this visual interface as a normal view into a window of its own and define navigation links.
Simple Web Dynpro Application Explained

Design User Interface in View Layout

- Design user interface using UI elements from the Web Dynpro Java UI element Library (around 100)
- Use drag and drop to position new UI elements in view layout
- See layout in WYSIWYG view designer
- See UI element structure in Outline view

Web Dynpro View Layouts

The Web Dynpro view provides a visual entity for defining a specific user interface part by arranging Web Dynpro UI elements inside the view layout. UI elements are the smallest UI entities in Web Dynpro. You can embed a view layout in either a window or in another view layout using the ViewContainer UI element.

Every Web Dynpro view is associated with its own view controller class implementing view-specific application logic like handling user actions or server-side events.
Simple Web Dynpro Application Explained
Declare Context Attribute to Store User Input

**View Controller Context**
- Web Dynpro-specific, hierarchical storage area for runtime data within each controller
- We define a single context attribute "Name" (of type string) to store the value entered by the user.

What is a Web Dynpro Context?

The *context* in the Web Dynpro programming model is a hierarchically structured storage place in a controller on server side. This storage place remains for the lifetime of the related controller and its components as Web Dynpro is included in the class of *stateful* Web framework. The context structure consists of elements called context nodes and context attributes.
Bind InputField to View Context

**Data Binding**
- Bind the “value” property of the InputField to context attribute “Name” in view context
- In this way the user input is automatically transported to the context at runtime
- The entered value can then be accessed via coding in the controller
Simple Web Dynpro Application Explained
Bind UI Element Event to Action

Action Binding
• Bind the event “onAction” of the Button UI element to action “Go” in view controller
• Every action is handled in its corresponding action event handler method inside the view controller class
Simple Web Dynpro Application Explained
Define Message Text in Message Pool

SUCCESS Message Text Definition
• Define (translatable) message in the message pool of your Web Dynpro component
• Define message key and message type

Based on the declared message text, the Web Dynpro Java Generation Framework (part of the IDE) automatically generates a corresponding Message constant in the IMessage<component name>.java interface to be accessed in Web Dynpro controller classes.
Simple Web Dynpro Application Explained
Implement Action Event Handler

**Invoke Message Manager**
- Report message text with the component's message manager service
- Access message manager via the generic component controller API

**Access Defined Message**
- Access defined and translatable message text via generated constant

**Access Context Attribute**
- Access context attribute Name via generated, typed context API

**Output on client side**

---

**What is a typed context API?**

Typed context APIs are context interfaces that are generated for the declared context nodes and attributes. For every declared context node `<node name>` a corresponding pair of typed context APIs, `I<node name>Element` and `I<node name>Node`, gets generated. Return types and parameter types of typed context API methods are not the generic Web Dynpro context APIs (IWDNode and IWDNodeElement), but the typed context APIs themselves.

**What is a user code area?**

For every Web Dynpro Java controller type (like window, component, custom or view controller) a corresponding Java class file `<controller name>.java` is created by the Web Dynpro Tools.

Within the **User Code Areas** of this controller class the application developer can implement the required custom code. All code and JavaDoc areas are delimited with the lines `//@begin` and `//@end`.

At the end of a controller class file the `//@begin others` - `//@end` code section can be used for any Java code that is not to be visible to other controllers/views or that contains constructs currently not supported directly by Web Dynpro such as inner classes, private methods or member variables etc. The content of this section is in no way managed or controlled by the Web Dynpro design time or the Web Dynpro Runtime.

It is important to understand, that the full functionality of a Web Dynpro controller is only partially implemented within the visible controller class comprising the application defined logic (**User Code Areas**). Many additional functions are implemented within the internal controller class and within the Web Dynpro Java Runtime itself. You can access these functions in your custom code via the special shortcut variables `wdThis` and `wdControllerAPI`. 

---
Controller Implementation

The complete implementation of a Web Dynpro controller is distributed across three different places (see next figure):

1. **Controller class**: Partially generated class comprising the application defined (self-implemented) logic or source code within user code areas. This is the only controller class an application developer implements source code. Application logic is implemented within component and custom controllers having no visual interface. Presentation logic is implemented within view and window controllers having a visual interface.

2. **Internal controller class**: Every Web Dynpro controller class is associated with an internal controller class named Internal<controller name>.java. This fully generated class acts as a mediator between the controller class and the Web Dynpro Java Runtime or other controller classes. It does not comprise any code implemented by the application developer but delegates to it.

3. **Web Dynpro Java Runtime implementation**: The Web Dynpro Java Runtime implements several generic controller APIs like IWDController, IWDCComponent, IWDViewController and IWDMWindowController. The generic controller APIs are required to access controller metadata (IWD...Info-APIs) or to modify controllers at runtime (e.g., dynamic event subscription). Furthermore they are needed to access some generic service-APIs like IWDMMessageManager, IWDMWindowManager or IWDTTextAccessor in your controller code.

Web Dynpro Java Runtime API
(See https://help.sap.com/javadocs/nwce/current/wdr/index.html)

Exposes generic runtime services implemented by the Web Dynpro Java Runtime to the application logic to be invoked inside Web Dynpro controllers: generic controller functions, message management, URL generation, resource handling, resource creation, NetWeaver Portal services or dynamic programming services like dynamic extension and modification of view layouts and contexts.

The Web Dynpro for Java API provides lots of APIs and service classes to minimize custom-coding efforts.
Web Dynpro Java Application Sample
Run Time View

SAP NetWeaver AS Java

run & interact

Web Dynpro Ajax Client

Client Abstraction Layer

Web Dynpro Java Runtime

Web Dynpro Ajax View Controller Instance

Component Instance

manage

manage

render

invoke

invoke

commu- nicate

Application Session

View Layout

User Code Areas

<<Mediator>>

Internal Controller Instance

A Very Simple Web Dynpro Application Sample

Congratulations, you made your first Web Dynpro Application works.

John Miller

Done

© SAP 2008 / SAP NetWeaver CE 7.11 EhP1 / Exploring Web Dynpro Java: Goals, Concepts, Benefits / Page 28
Web Dynpro Phase Model and the Hollywood Principle

Answer

Web Dynpro applies the “Hollywood Principle”. In other words: “don't call us, we'll call you.” This means Web Dynpro calls **hook methods** implemented in your controller classes in a pre-defined order.

This order is defined by the **Web Dynpro phase model** which is processed in every request-response cycle for all active components.

The **hook methods** provide a static interface between your business application and the Web Dynpro Framework – no matter what functionality your application contains.

The Hollywood Principle

“The Hollywood principle is a software design methodology that takes its name from the cliché response given to amateurs auditioning in Hollywood: "Don't call us, we'll call you". It is a useful paradigm that assists in the development of code with **high cohesion** and **low coupling** that is easier to debug, maintain and test. ... Instead of your program running the system, the system runs your program. .... All of these mechanisms require some cooperation from the developer. To integrate seamlessly with the framework, the developer must produce code that follows some conventions and requirements of the framework.”

This very simple Web Dynpro application demonstrated lots of goals, concepts and benefits:

- **Declaration-centric development process**: Even more than 80% of all development steps were based on modeling and declaration. We only implemented 3 lines of Java code!
- **Code Generation**: All controller classes (we only used the view controller) and interfaces (we only used the message pool interface) were generated based on our declarations.
- **Automatic Data Transfer** between client and server based on data binding.
- **Automatic session management** done by Web Dynpro Java Runtime.
- **Client device independence**: Application runs in Web Dynpro Ajax Client on Firefox and Microsoft Internet Explorer.
- **Internationalization**: Our reported message text defined in the message pool can be translated to other locales. At runtime the Web Dynpro Runtime environment automatically loads locale-dependant text from corresponding translated resource bundle.
Exercise 1
Exploring a Very Simple Web Dynpro Java Application
Exploring Web Dynpro Java: Goals, Concepts, Benefits

1. Web Dynpro Basics at a Glance
2. Programming Model
3. User Interface
4. Component Model
5. Backend Connectivity
6. Web Dynpro Flavors and Differentiators
Web Dynpro Java UI Technology

Client Abstraction

Client abstraction for the unified design of user interfaces regardless of a specific client (device) technology

Generic UI services for recurring application parts, such as input helps, type validation, error messages, accessibility or implicit personalization

Consistency of the user interface and a unified look and feel by providing a comprehensive library of user interface elements with in-built AJAX function

Multilingual capability and translatability by removing language-dependent texts and automatic text filling according to the user locale
Web Dynpro Client Abstraction and Client Device Independence

Web Dynpro Client Device Independence

Run the same application on different client devices and browsers **without code modification**. Device and Browser recognition with serverside rendering is all done by the Web Dynpro Java Runtime.

Supported Web Dynpro Mobile Client Devices:

**Pocket PC**
- Support: LAN, WLAN, Bluetooth, GSM or GPRS
- Browsers: Pocket IE

**Blackberry**
- Support: Software of Blackberry and MDS should be 3.6 or higher
- Connection: GPRS or WLAN
- Browsers: Blackberry Browser

**Nokia Series 80**
- Connection: GPRS or WLAN
- Browsers: Opera Browser

**Nokia Series 60**
- Support: From 3rd Edition Feature Pack 1
- Connection: GPRS or WLAN
- Browser: S60 OSS Browser
Closely connected with client abstraction is the well-stocked library of more than 90 Web Dynpro UI elements that you can use to develop Web user interfaces that appear consistent and include a significant level of user interaction. Tables, trees, context and pop-up menus, toolbars, or calendars are also included here.

| UI Element            | Web Dynpro also relieves the application developer of the difficult task of adapting the user interface (called view layout) to different clients. This is achieved through an interface element description (such as properties and events) that is abstract and separate from specific client technologies and even from certain end devices (such as desktop PCs or mobile devices). This technique is also known as client agnostic UI. This means that the application developer constructs the user interfaces without knowledge or presumption of the client technology to be used at runtime. Client technologies here include the different browser clients such as Mozilla Firefox or Microsoft Internet Explorer of which there are multiple versions. A Web Dynpro developer only has to embed and define user interface elements in view layout and at no time has to deal with implementing HTML, CSS, or JavaScript code. At runtime, Web Dynpro itself renders the user interface for different Web Dynpro clients on server side, such as the Web Dynpro Ajax client (Browser client), the Web Dynpro Flex client, or the Web Dynpro client for mobile devices. The Web Dynpro Flex Client is first delivered with SAP NetWeaver CE 7.1 Service Pack Stack 3. |
What about writing HTML or JavaScript Code in Web Dynpro Java?

**Question**
Does Web Dynpro Client Abstraction imply that my developers are not able to write any HTML, JavaScript or Ajax code?

**Answer**
Definitely yes! The developer is only concerned with writing Java code.

**No HTML, JavaScript, browser or AJAX knowledge is needed in Web Dynpro Java.**

This makes the Web Dynpro Java framework much easier to learn and the code better to maintain.

Client specific HTML is automatically rendered on the server for each supported client device. JavaScript code is contained in the UI element Library and gets sent to the client on-demand.

Ajax functions and *XMLHttpRequest*-based client-server communication (e.g. for table paging, asynchronous input validation or suggestion) are fully implemented by Web Dynpro.
Custom UI Element Development Not Possible

Answer

No, it is not possible for application developers to develop their own UI elements due to the fact that the new UI element would have to support:

- Client device independence
- Accessibility
- Personalization support
- Right-to-Left support etc...

SAP continuously enriches the Web Dynpro UI Element Library with new elements and functions like RowRepeater, ContextMenu, Drag&Drop, Calendars or FunctionKey in NW CE 7.1.

To fulfill special UI requirements beyond the existing UI element library, like specific graphical highly interactive UI elements, you can apply the new Adobe Flash Island for Web Dynpro Java technique first available with SAP NetWeaver CE 7.11 EhP1.

Question

Is it possible to extend the Web Dynpro UI Element Library shipped within SAP NetWeaver with own “custom” UI elements?

On the one hand, a client abstraction or client device independence of this type makes it possible to call the same Web Dynpro application without modifications on different end devices and with different client programs, which of course is a great advantage and significantly reduces the development costs.

On the other hand, it must be pointed out that Web Dynpro interface elements can only be changed as far as the supported clients and requirements for international business applications allow. This means, for example, that you cannot specify the pixel-precise position of interface elements or develop any additional interface elements of your own (because texts are different lengths in different languages and font sizes, interface elements are aligned automatically by the Web Dynpro runtime environment making pixel-precise positioning by the application developer unnecessary).

This difference of Web Dynpro should not, however, be seen as a disadvantage, but rather it should be understood more as the price to be paid for the considerable advantages of client independence with Web Dynpro.
Web Dynpro AJAX Client Succeeds HTML Client in SAP NetWeaver CE 7.11 EhP1

The Web Dynpro AJAX Client … is the renamed successor of the former Web Dynpro HTML Client based on the following Ajax functions implemented in NetWeaver CE 7.11 EhP1:

- **Drag & Drop, Suggestion, Page Rearrangement**
- **Use of XMLHttpRequest**
- **Asynchronous Input Validation**
- **InputField UI elements supports onChange event, In-place editing and input prompt**
- **Extensive Keyboard Support** (hot and access keys)
- **MessageBasedTrigger UI element**
- **Fluid animations based on UR Lightspeed**
- **Enhanced client/server communication supporting on-demand JavaScript**
- **New Web Dynpro UI theme SAP Tradeshow Plus**
In-Place Editing

Drag & Drop

Hot & Access Key Support

Suggestion
New SAP Signature Theme …
More Openness in Web Dynpro Java – Adobe Flash Islands for Web Dynpro Java

In SAP NetWeaver CE 7.1 EhP1 Web Dynpro Java provides openness to other UI technologies like Adobe Flash or Java Server Faces.

Adobe Flash Islands for Web Dynpro Java

- Support rich user interaction with Adobe Flash applications utilizing transition effects, drag & drop, sliders, tooltips etc.
- Provide a FlashIsland UI Element to integrate additional Flash Controls like YahooMaps in Web Dynpro view layouts.
- Provide subset of WD4VC Analytics UI Elements like the AnalyticsChart UI Element in Web Dynpro Java UI Element Library.

Implications of using Adobe Flash Islands for Web Dynpro Java

- You will have high development and maintenance efforts (over a long time)! Extremely high even for experts. Specific UI programming skills needed. Complex and difficult debugging (multi: system, development environment and languages)
- You will have to care for the SAP Standards on your own!
  - Accessibility
  - Internationalization: Provide translated text on the server side and transport them to the client. Provide R2L flex UIs in addition to L2R or alternative implementation. Current limitation for Japanese due to Flash Player bug
- Performance: Do not use more than 3 Flash Islands on the screen
  - Avoid data aggregation on client side
  - Provide suitable paging of data
- You will have to provide “standard” Web Dynpro Java features on your own!
  - Value help and field help
  - Personalization
- You will have to work within a weaker development infrastructure.
Web Dynpro Openness and Integration Capabilities 1/1

Leverage rich variety of integration capabilities in Web Dynpro Java applications:

- **SAP NetWeaver Portal Integration:**
  Enhance a Web Dynpro application by seamlessly integrating it into SAP NetWeaver Portal to surmount the boundary between the application layer (Web Dynpro) and the portal layer (SAP NetWeaver Portal)*. Use Portal services like WDPortalWorkPortectMode, WDPortalEventing or WDPortalNavigation.

- **Adobe Integration:**
  Integrate Adobe Interactive Forms into Web Dynpro Java UIs

- **Adobe Flash Island Integration:**
  new in SAP NetWeaver CE 7.11

- **Office Integration:**
  Open, Edit and Save MS Office Word and Excel files in-place in your running Web Dynpro Java application UI

- **Suspend/Resume navigation to externally running business applications:**
  Navigate to and interoperate with external applications while keeping your Web Dynpro Java application running in suspend mode. Navigate back, resume Web Dynpro Java application and use retrieved data in controller logic.

* From article „Integrate Web Dynpro into your SAP NetWeaver Portal to create dynamic, flexible applications“, Jochen Guertler, SAP Professional Journal, Jan/Feb 2008, Volume 10, Issue 1
Leverage rich variety of integration capabilities in Web Dynpro Java applications:

- **Business Graphics:**
  Automatically generate graphics from Web Dynpro context data with BusinessGraphics UI element and Internet Graphics Server (IGS).

- **Geo Maps:**
  Integrate Maps from a Geographic Information System with GeoMap UI Element

- **JNet/JGantt:**
  Integrate network graphics as active components with JNet and JGantt

- **JSF Integration:**
  Integrate JSF controls inside Web Dynpro Java user interface

**NOTE: Restricted Usage in NW CE 7.1 EhP1 (see next slide)**

** Regarding GeoMap Integration see SAP Notes 994568 (GIS Implementation) and 1135220**
More Openness in Web Dynpro Java
JSF Integration

In SAP NetWeaver CE 7.11 EhP1 Web Dynpro Java provides openness to other UI technologies like Adobe Flash or Java Server Faces with new …

Web Dynpro JSF Integration*

- Allow usage of JSF (composite) UI components in Web Dynpro UIs.
- UI Mashups with Web Dynpro and JSF: combine UI content from both side to allow a novel form of reuse.

NOTE:

In NW CE 7.1 EhP1 the new Web Dynpro JSF Integration technique can only be applied for evaluation purposes. Customers can only use it based on a SAP 90 DAYS EVALUATION LICENSE AGREEMENT.

WebDynproJsfBridge UI Element:

This UI element represents an area on a Web Dynpro page where the displayed content is rendered by an embedded JSF UI component. The WebDynproJsfBridge UI element can contain 0..1 JSF component. The Web Dynpro tools provide support for importing JSF components, adding one to a WebDynproJsfBridge, setting and binding the component properties etc. The component can be a composite component (e.g. HtmlDataTable component containing several HtmlColumn components). In this case the tools only provide the handling of the root component (properties etc.) but it is possible to handle the nested components programmatically via the WebDynproJsfBridge methods (e.g. binding of properties of nested component).
Data Binding Between UI and Context Elements

All UI elements have a set of properties. These properties can obtain a value in one of two ways. Either:

- From a value hard-coded into the property field, or
- By being bound to a node or attribute in the view controller’s context

Data Binding = binding UI element properties to view context elements

Decouples UI elements from the application logic implemented inside controller classes.

When the properties of a UI element are bound to attributes in the context, there is an automatic, two-way flow of data between the UI and the context state.

UI elements in Web Dynpro are abstract representations of the actual UI elements that are rendered on the client device.

The reason for this is to achieve client device independence during the development process.

With very few exceptions (E.G. id and layoutData), all UI element properties can be bound to a context node or attribute.
Adapting the Design of UI Element Containers with pre-defined design Property Values

ToolBarPopin TextBar

ToolBarPopin Content Text

TextView UI element as children[0] of Tray UI element

ToolBarPopin TextBar

ToolBarPopin Content Text

TextView UI element as children[0] of Tray UI element

© SAP 2008 / SAP NetWeaver CE 7.11 EhP1 / Exploring Web Dynpro Java: Goals, Concepts, Benefits / Page 47
Adapting the Design of UI Element Containers with pre-defined design Property Values

Tray design = FILL
ToolBarPopin design = PLAIN
ToolBar design = EMPHASIZED
TextView (ToolBarPopin Content) semanticColor = POSITIVE

WDTrayDesign and WDPopinDesign Enumerations

The design property of a Tray or ToolBarPopin UI element can have one of the following three predefined values:
1. FILL: The content area has a background color.
2. PLAIN: The content area has a white background and borders.
3. TRANSPARENT: The background is transparent, the content area has no borders.

WDTToolBarDesign Enumeration

The design property of a ToolBar UI element can have one of the following three predefined values:
1. EMPHASIZED: Non-standard background color.
2. STANDARD: Standard background color (default)
3. TRANSPARENT: Transparent background.

WDTextViewSemanticColor

Sets the semantic color in a TextView control.
1. CRITICAL TextView color representing critical state, e.g. red
2. DIMINISHED TextView color representing diminished state.
3. MARKED1 TextView color representing a first marked state.
4. MARKED2 TextView color representing a second marked state.
5. NEGATIVE TextView color representing negative state, e.g. orange
6. POSITIVE TextView color representing positive state, e.g. green.
7. STANDARD TextView color representing standard color.
Adapting the Design of UI Element Containers with pre-defined design Property Values

Tray design = PLAIN
ToolBarPopin design = FILL
ToolBar design = TRANSPARENT
TextView (ToolBarPopin Content) semanticColor = NEGATIVE
Web Dynpro UI Element Positioning and LayoutData

UI Element positioning is largely controlled by the container layout they belong to. But some things can be controlled at the UI element level.

- The container layout that the UI Element belongs to dictates what kind of LayoutData (properties) that UI element has.

```
RowLayout
- RowHeadData
- RowData

GridLayout
- GridData

ColumnLayout
- ColumnHeadData
- ColumnData

FlowLayout
- LayoutData

MatrixLayout
- MatrixData
```

**Note:** Elements belonging to RowLayout only have the shown properties if their layoutdata property is set to RowHeadData.
IMPORTANT NOTE:

Although the LayoutData Properties are listed on UI element level they refer to the **virtual cell** containing the UI element, but not to the UI element itself.

The LayoutData defines **cell specific** layout properties (like width, height, vertical gutter or paddings) and how a UI element is laid out in its embedding container cell, i.e. on the container grid (vertical alignment, horizontal alignment).
Why Web Dynpro Java Does not Support Pixel-Perfect Design

Answer

Web Dynpro UI elements will change size as a result of localization, varying font sizes and even dictionary type metadata (field lengths).
E.G. Words change length after translation!
Furthermore your Web Dynpro application can run on different clients that calculate box sizes (such as widths, borders, or margins) using client-specific algorithms and therefore client-specific results.

Question

Why does Web Dynpro Java not support Pixel-Perfect Alignment of UI elements?
(also called “Pixel-Perfect Design”)

"Web Dynpro has been designed to support multiple languages automatically, and does not require the developer to write language-specific coding.

If Web Dynpro were to allow the developer to position UI elements exactly on the screen, then such information would immediately become language specific because of the simple fact that words change length (and possibly direction) when translated from one language to another.
Translation is an awkward enough process as it is without adding UI element placement into the mix.Therefore, SAP took the deliberate design decision to have all UI elements positioned automatically by the Web Dynpro Framework."

From: „How to avoid Web Dynpro Java implementation nightmares”, Chris Whealy, SAP Professional Journal, March/April 2008

Example Description

•Intrinsic sizing of UI elements automatically extends the Label width and correctly positions the InputField
•Pixel-perfect positioning of InputField would truncate the Label
Web Dynpro Java provides several in-built or generic UI services to rapidly fulfill standard UI requirements and services

- **Navigation and View Management**
- **Valuehelp Services:**
  Search Keys and display texts in Simple type enumerations (EVS, SVS), search field values in business objects (OVS)
- **User Input Validation:**
  validate user input based on data type definitions
- **Message Reporting:**
  Easily report message texts on the user interface. Automatically mark and highlight erroneous fields.
- **Accessibility:**
  render user interfaces fulfilling section 508 and Web Content Accessibility Guidelines (WCAG) requirements
Generic UI Services 2/2

Web Dynpro Java provides several in-built or generic UI services to rapidly fulfill standard UI requirements and services

- **Internationalization:**
  Allow translation of Web Dynpro applications without code modification and run them with different user locales. Support Right-To-Left rendering (RTL-Support).

- **Modification-free UI Adaptation/Personalization:**
  Personalization of Web Dynpro applications running inside SAP NetWeaver Portal defined by administrator (role and user-based) or end-user.

- **Adaptability and Custom Extension Field Support**: 
  add new fields in dictionary append structures and run application with extended user interface (propagate added field to the UI) but without code modification

*Field Extensibility in Web Dynpro Java is only supported in Adaptive RFC Models 1 and 2 (part of NetWeaver CE 7.11 EhP1) calling Remote Function Modules in an SAP system.*
Exercise 2
Using the AnalyticsChart UI Element
Exploring Web Dynpro Java: Goals, Concepts, Benefits

1. Web Dynpro Basics at a Glance
2. Programming Model
3. User Interface
4. Component Model
5. Backend Connectivity
6. Web Dynpro Flavors and Differentiators
The Web Dynpro UI Technology
Component Concept

Loose component coupling using Component Interface Definitions (strategy and dependency injection patterns)

Leverage component-based architecture in Web Dynpro Java business applications for sake of reuse, better maintainability, extensibility and flexibility

Automatic framework-controlled component lifecycle management

Component model for logical separation, loose coupling, and the reuse and team-based development of application components

Separation of concerns (SoC) with Web Dynpro Java Components

This process breaks down a computer program into distinct features that overlap in functionality as little as possible. Any piece of interest or focus in a program is a concern. Typically, concerns are synonymous with features or behaviors. Traditionally, modularity and encapsulation have been used to progress toward SoC with the help of information-hiding. In Web Dynpro, modularity and encapsulation are achieved with Web Dynpro components and information hiding with their visual and programmatic interfaces.
The Web Dynpro component is the Main Building Block in Web Dynpro Development

The **Web Dynpro component** is considered as the most important, integral building block for developing Web Dynpro business applications.  
- Web Dynpro components are the only places into which you can put Web Dynpro functionality.  
- The architecture of a Web Dynpro component is based on the Model-View-Controller (MVC) model, which establishes a clear separation between business data model, user interface, and controller implementation or application logic.

**Web Dynpro Component Anatomy**

A Web Dynpro component mainly consists of
- different **programmatic controller entities** that implement the application and navigation logic (such as component, custom, window and view controller classes)
- **visual entities** that implement the user interface part (such as interface views, windows, view layouts, and UI elements).

Additionally the inner part of a Web Dynpro component implements the visual and programmatic interfaces exposed to other components outside:
- **Component interface view** to expose the application user interface implemented inside a visual Web Dynpro component for external use to build modularized UIs
- **Component interface controller** for programmatic communication (method invocation and eventing) and data transfer purposes (interface context mapping) between different Web Dynpro components

Several very important differences between Web Dynpro development and other MVC based toolsets:

In Web Dynpro:
- The Component is the fundamental unit of development, not the web page.
- A component is not a single Java class. Instead, it is an aggregation of independent, but interrelated classes.
- No individual class within a Web Dynpro Component can function independently from the rest of the component.
- The unit of reuse in Web Dynpro is an entire component, **not** an individual view or controller within a component.
Think in Web Dynpro Components To Modularize Web Dynpro Business Applications

Leverage component-based architecture to build state-of-the-art Web Dynpro Java business applications.

With component-oriented Web Dynpro Development architects and developers can achieve component reuse, better maintainability, flexibility and understandability to reduce **Total Cost of Development** (TCD).

Web Dynpro components are the building blocks for developing Web Dynpro applications. A component-based architecture makes it possible to split complex applications into individual, functionally separated components and then to develop, maintain, and later enhance them separately with a clear division of tasks. This is possible due to some fundamental features of the Web Dynpro component model, such as its interface concept, its capability for defining component interfaces abstractly, its ability to modularize large applications, and the facility for reusing Web Dynpro components.

**Component Benefits**

- Break down applications into manageable parts.
- Distribute work between developers
- Stay independent of implementation details (black box approach). Components hide implementation behind an interface.
- Construct reusable entities component which can be used in several other components
Web Dynpro Java Development is Fully Integrated into the SAP NWDI

Development and Lifecycle Management of Web Dynpro Java applications is fully integrated into the **NetWeaver Development Infrastructure** with its own Component Model (Products, Software Components and Development Components).

Web Dynpro Java Applications are mainly stored and separated in **Development Components** of type **Web Dynpro**.

Large-Scale, Component-Based Web Dynpro Applications are developed within the NetWeaver Development Infrastructure (NWDI). Reasons …

The NWDI provides the infrastructure for fulfilling Web Dynpro development needs:
- Break down applications into manageable parts
- Distribute work between multiple developers
- Construct reusable entities with Web Dynpro Components

The NWDI provides several services:
- Design Time Repository (DTR)
- Component Build Service (CBS, Build plugins)
- Change Management Service (CMS)
- Translation Tools

“The development according to the component model is organized by development configurations, which define the development landscape for a specific software development project. Development configurations can be created locally (instead of using the SAP NetWeaver Development Infrastructure).

If you want to use local development configurations for team development additionally, central processes can be set up with the command line tool delivered with the Developer Studio for building the application centrally and packing it as a software component archive (SCA) for delivery. Optionally you can enhance these processes with external (non-SAP) infrastructure.”

Component Interface Definitions for Loose Web Dynpro Component Coupling

You can define the Web Dynpro component interface outside a concrete Web Dynpro component implementation in an abstract way.

Component Interface Definition Strategy

At design time, loosely couple a parent component with its child component using a component interface definition, which has no implementation inside.

At runtime, programmatically create a component instance implementing this component interface definition.

(See „Strategy“ and „Dependency Injection“ Design Patterns)

For more details on this topic, see the “Web Dynpro Component Interface Definitions in Practice” Learning material on SDN:

Component-Based Architecture in Web Dynpro Java Business Applications

The functionality of Web Dynpro components allows you to divide a large-scale business application into smaller, more manageable parts, so that different concerns are separated into logical units:

- Split rich user interfaces into several visual components executing different parts of the application UI
- Centralize access to the business logic in faceless model components implementing no user interface
- Build one root component bundling together all other components
- Build reusable components for recurring parts of your business applications

Apply Componentization Patterns

- Share context data across component borders with interface context mapping
- Visually embed component interface views
- Delegate application logic with interface method invocation and eventing
SAP Professional Journal 2008 – Article Series

“Leveraging component-based architecture in Web Dynpro for Java business applications”

by Bertram Ganz (Senior Product Specialist, SAP AG)
and Richard Tucker (Principal Web Development Architect, Atos Origin UK)

Part 1 — Design methodology, concepts, and case study (SPJ, May/June 2008)
Part 2 — Component models (SPJ, July/August 2008)
Part 3 — Componentization patterns in practice (SPJ, September/October 2008)
Agenda

Exploring Web Dynpro Java: Goals, Concepts, Benefits

1. Web Dynpro Basics at a Glance
2. Programming Model
3. User Interface
4. Component Model
5. Backend Connectivity
6. Web Dynpro Flavors and Differentiators
The Web Dynpro Java UI Technology

Model Abstraction

Model abstraction (based on a defined standard called Common Model Interface (CMI)) for unified connectivity of a business application to different backend access technologies such as RFCs, Web services, Enterprise Services or Enterprise JavaBeans.

Automatic data transport between Web Dynpro frontend server, and backend server based on model binding concepts.

Separation of Business Logic and Presentation Logic.
How Web Dynpro Realizes Backend Access

Web Dynpro Model

**Answer**

A Web Dynpro Model is a generated entity which includes an aggregation of model classes needed to connect a Web Dynpro component to business logic contained in a back end system.

Depending on the concrete backend technology used (whether it be RFC, Web service, or EJB 3.0) the Web Dynpro developer imports either

- **Adaptive RFC model**
  - To call a remote enabled ABAP module

- **EJB 3.0 model**
  - To call a stateless EJB 3.0 session bean

- **Adaptive Web service model**
  - To call a Web service or Enterprise service.

All model classes generated automatically and all present the same standardized interface known as the **Common Model Interface**.

See JavaDoc for the Common Model Interface on SAP Online Help:
How Web Dynpro Realizes Backend Access
Context-To-Model Binding

Question
And how is an imported Web Dynpro Model used to connect a Web Dynpro Java application to the business logic?

Answer
Next, the application developer creates a corresponding data structure from the imported data model in the controller context known as context-to-model binding.

Connection management, communication and data transfer between an application and the backend system is executed by the generated model implementation and the Web Dynpro runtime environment.

The context structure is used in a controller to bridge the gap between the user interface and the model and to be able to define the automatic data transport based on the data binding, context mapping, and model binding.
How Web Dynpro Realizes Backend Access
Data Transport and Model Execution

Answer

The context structure created in the **context-to-model binding** declaration step only represents the data model itself and is independent from the underlying backend technology.

The context structure is then used to distribute the data within the Web Dynpro component:

- **Model binding**: Context to model
- **Context mapping**: Controller to Controller
- **Data binding**: UI to context

This means that as a result, the implementation effort for the application developer is reduced to triggering backend calls in the controller implementation.

Data transport within the component is fully automated by the Web Dynpro Java runtime.

Question

But how does the data flow from the context to the user interface work?

Does the Web Dynpro developer have to do anything to implement this?
How Web Dynpro Realizes Backend Access

Context Mapping

Answer

A Web Dynpro Component is built from independent, yet inter-related controllers. These controllers would normally not share information with each other.

Context mapping is the primary mechanism for sharing data between controllers within the scope of a single component. (Context mapping can also be used to share data across component boundaries).

Data stored in the component controller can be referenced by other controllers in the component without the need to copy data.

Using context mapping, business data stored in a model can be easily used in multiple contexts without the need for programming.

Question

I know the Web Dynpro data binding concept for automatic data transport between UI and context. But what do you mean with context mapping?
Analyzing the Binding Chain from UI To Model

1. **Data Binding UI to Context**
2. **Context Mapping Context to Context**
3. **Model Binding Context to Model**

Binding Chain:

1. **StudentUI**
2. **EJBModel**
3. **Student**
Executing a Model Object In Controller Class
To Communicate with Backend

```java
//@@begin javadoc:findAllStudents()
/**
 * Method declared by application.
 */
//@@end
public void findAllStudents() {
    //@@begin findAllStudents()
    try {
        wdContext.currentRequest_Students_findAllElement().modelObject().execute();
    } catch (EJBModelExecuteException e) {
        wdComponentAPI.getMessageManager().reportException(
            e.getLocalizedMessage());
    }
    //@@end
}
```

Controller Context

- Request_Students_create
- Request_Students_findAll
- Response
  - return
  - address
  - age
  - firstname
  - id
  - lastname

Web Dynpro Custom Controller Class
Exercise 3
Using an EJB Model in a Web Dynpro Java Application
Agenda

Exploring Web Dynpro Java: Goals, Concepts, Benefits

1. Web Dynpro Basics at a Glance
2. User Interface
3. Programming Model
4. Component Model
5. Backend Connectivity
6. Web Dynpro Flavors and Differentiators
Two Flavors of Web Dynpro
Web Dynpro ABAP and Web Dynpro Java

Answer

Web Dynpro in general is SAP’s standard UI technology for developing business applications that have Web-based user interfaces.

It consists of a runtime environment and a graphical development environment with special tools that are integrated into either the ABAP or Java development environments.

They both share the common Web Dynpro programming model concepts but feature partially different functions in different NetWeaver releases. More details will be described on the next slides.

Note:

There is no tool to convert a Web Dynpro ABAP application meta-model (except the controller coding) into a Web Dynpro Java meta-model or vice versa.

Question

SAP ships Web Dynpro in two flavors: Web Dynpro Java and Web Dynpro ABAP. Can you tell us some more details on this?

What is Web Dynpro ABAP?

ABAP (Advanced Business Application Programming) is SAP’s object oriented, proprietary programming language for developing business applications within an SAP system. Web Dynpro ABAP (WDA) is the main SAP standard UI technology for developing Web application UIs in the ABAP environment. It consists of a runtime environment and a graphical development environment with special tools that are completely integrated into the ABAP development environment - the ABAP Workbench (transaction SE80):

This full integration with the ABAP development environment provides direct access to ABAP data, business logic, and to Data Dictionary Search Helps. To align and support the development of the ERP User interfaces, a Floorplan Manager (FPM) for WDA framework is also available within the ABAP stack. The FPM provides central implementation of floor plans (Object Instance Floorplan, Quick Activity Floorplan, and Guided Activity Floorplan), support for WDA screens integration, and tools to adapt and configure the applications by the customers. WDA has been available since October 2005 (SAP NetWeaver 7.0 (2004s) – Web Application Server 7.0) and is widely used by the SAP Business Suite and will continue to be the UI technology of choice for mainstream Business Suite applications (ERP, PLM, SCM, SRM, FIM, HCM).

FPM is available since WDA 7.00 SP13 and is used by all ERP projects using WDA as of SAP EhP4 for SAP ERP 6.0.

Additional information about WDA can be found here: https://www.sdn.sap.com/irj/sdn/nw-ui?rid=/webcontent/uuid/80aef577-543f-2a10-d19c-d83a565efe37#section2

From: SAP NetWeaver Web Dynpro ABAP & Java – FAQ, Dafna Yanay, SAP AG, SDN 2008
Web Dynpro Java and ABAP have many similar capabilities for creating application UIs, but also unique capabilities and differences.

As depicted by the diagram above, both tools create web-based business application UIs and use the MVC design patterns to support a strict separation of business and presentation logic. In addition, they both will support (in EhP1) integration with high interactivity islands to enrich the basic Web Dynpro UI offering with pixel-perfect high, animated, and improved UIs. Naturally, there are also differences, mainly because WDJ is implemented in Java and offers an Eclipse-based development environment, while WDA is implemented in ABAP and completely integrated into the ABAP stack. With WDJ, developers can create their UI applications based on various data sources such as enterprise services or EJBs and attach these UIs to an end to end business process using the new BPM (Business Process Modeling) tool (will be available with EhP1 of CE). On the other hand, when using standard business logic from the SAP operational systems it is much easier to access it using WDA and consume the data directly from the ABAP system via native ABAP coding. In addition, WDA offers a MIME repository to manage and use different objects such as graphics and icons. The Web Dynpro application runtime and the UIs’ look and feel are also similar since the rendering of the different UI controls in both flavors is provided by the same Unified Rendering (UR) layer. This approach guarantees the same unified look and feel for all Web Dynpro user interfaces independent from the implementation language, together with additional services like accessibility, personalization and right to left rendering.

Although the UR concept promises the same UI controls offering in both flavors, due to resource and release schedule reasons some UI element implementations (Calendar and Column layout) are currently missing in the WDA flavor and vice versa, the AcfExecute and AcfUpDownload are only implemented in WDA. In the coming release of both WDJ (EhP1 of CE) & WDA (Ehp1 of NW7.0), the rendering frameworks are improved and use Ajax technology to provide a better user experience. (The new framework will provide AJAX functions like: suggestion support, in-place editing, asynchronous input-validation for numeric fields, enhanced keyboard support and more).

From: SAP NetWeaver Web Dynpro ABAP & Java – FAQ, Dafna Yanay, SAP AG, SDN 2008
Web Dynpro is SAP’s strategic UI programming model, which offers two programming languages for creating all kinds of business applications:

- **WDA** which is the main SAP standard UI technology for developing Web application UIs in the ABAP environment. This flavor is widely used by the SAP Business Suite to create the mainstream Business Suite applications.

- **WDJ** which is a standard, open source-based Eclipse framework UI technology for developing Web application UIs in the Java environment. This flavor is heavily used by System Integrators and ISVs, who innovate on top of the existing SAP applications and create new processes using open software solutions and services such as enterprise services and other Java components.

Since the main differences between the tools are due to the different programming languages, the decision on which flavor to use should first and foremost be based on the available development resources, location of business data/processes and existing infrastructure.

In addition to these major differences there are also lots of technical and functional implementation differences. These differences are temporary, and in general should be implemented in both tools but currently implemented only in one of them due to resources and release schedule reasons. ...

The decision on which tool to use shouldn’t be based only on these differences. Comparing both Web Dynpro flavors on a functional level is highly time-consuming and very technical. In many cases the differences only affect special mini-features. Nevertheless a mini-feature can sometimes exactly hit a specific application requirement that you need (e.g. mobile access is supported only with WDJ).

**From:** *SAP NetWeaver Web Dynpro ABAP & Java – FAQ, Dafna Yanay, SAP AG, SDN 2008*
Answer

WD4VC is the runtime environment for Visual Composer for Composition Environment (CE) 7.1 applications. Technically, it is an extension of the Web Dynpro Java Runtime environment.

WD4VC can be used to rapidly develop business applications consuming e.g. eSOA -and BI-services without having in-depth Java knowledge:

- **Easy-to-Use**: fully model-based, no coding required
- **BI Connectivity**: out-of-the-box support for BI data services using the BI consumer service layer.
- **ALV-Table support**
- **Complex data structure support** (consumption of eSoa services)
- **Web Dynpro component integration**

Question

And what about the third flavor named Web Dynpro for Visual Composer (WD4VC)?

What is Visual Composer?

Visual Composer is a graphical modeling environment designed for the rapid creation of web-based user interface applications. Visual Composer has been designed to enable people without traditional programming skills to create enterprise application UIs using standardized components that meet SAP standards and quality criteria. The latest version of Visual Composer (Visual Composer for CE) also introduces new tools for creating portal content (worksets, roles, pages, and iViews), embedding BI data and creating Voice applications from the Visual Composer modeling environment.

From: “SAP NetWeaver Visual Composer and Web Dynpro Java – FAQ UI and Modeling recommendations with CE”, *Dafna Yanay, SAP AG, SDN 2008*
Comparison Between WD and Visual Composer  
Design Time Aspects + Programming Model

**Web Dynpro**
- Model & code-based development environment (Eclipse-based)
- Powerful combination of Java and MVC paradigm
- Openness to various data sources using the CMI model

**Visual Composer**
- Creation of web-based, user interfaces for business applications
- Separation of presentation logic and data logic
- Componentization/reuse
- Access to different backend systems
- Model – based development environment
- Fast learning curve & low entry barrier (non-programmer can read and develop)
- Rapid application development

© SAP 2008 / SAP NetWeaver CE 7.11 EhP1 / Exploring Web Dynpro Java: Goals, Concepts, Benefits / Page 78
Comparison Between WD and Visual Composer

Runtime Aspects

Web Dynpro
- Large set of UI controls

WD for VC
- Personalization
- Client agnostic
- Accessibility
- Translation
- UI Rendering (common look & feel, supported browsers, etc.)
- Mobile support
- Adobe interactive forms support
Further Information

SAP Public Web:
- SAP Developer Network (SDN): [www.sdn.sap.com](http://www.sdn.sap.com)

Related SAP Education and Certification Opportunities
No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

SAP, R/3, xApps, xApp, SAP NetWeaver, Duet, SAP Business ByDesign, ByDesign, PartnerEdge and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned and associated logos displayed are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

The information in this document is proprietary to SAP. No part of this document may be reproduced, copied, or transmitted in any form or for any purpose without the express prior written permission of SAP AG. This document is a preliminary version and not subject to your license agreement or any other agreement with SAP. This document contains only intended strategies, developments, and functionalities of the SAP® product and is not intended to be binding upon SAP to any particular course of business, product strategy, and/or development. Please note that this document is subject to change and may be changed by SAP at any time without notice. SAP assumes no responsibility for errors or omissions in this document. SAP does not warrant the accuracy or completeness of the information, text, graphics, links, or other items contained within this material. This document is provided without a warranty of any kind, either express or implied, including but not limited to the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.

SAP shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials. This limitation shall not apply in cases of intent or gross negligence.

The statutory liability for personal injury and defective products is not affected. SAP has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third-party Web pages nor provide any warranty whatsoever relating to third-party Web pages.

Die in dieser Publikation enthaltene Information ist Eigentum der SAP. Weitergabe und Vervielfältigung dieser Publikation oder von Teilen daraus sind, zu welchem Zweck und in welcher Form auch immer, ohne die ausdrückliche schriftliche Genehmigung durch SAP AG nicht gestattet. In dieser Publikation enthaltene Informationen können ohne vorherige Ankündigung geändert werden.

Einige von der SAP AG und deren Vertriebspartnern vertretene Softwareprodukte können Softwarekomponenten umfassen, die Eigentum anderer Softwarehersteller sind.

SAP, R/3, xApps, xApp, SAP NetWeaver, Duet, SAP Business ByDesign, ByDesign, PartnerEdge und andere in diesem Dokument erwähnte SAP-Produkte und Services sowie die dazugehörenden Logos sind Marken oder eingetragene Marken der SAP AG in Deutschland und in mehreren anderen Ländern weltweit. Alle anderen in diesem Dokument erwähnten Namen von Produkten und Services sowie die damit verbundenen Firmenlogos sind Marken der jeweiligen Unternehmen. Die Angaben im Text sind unverbindlich und dienen lediglich zu Informationszwecken. Produkte können länderspezifische Unterschiede aufweisen.


Alle Rechte vorbehalten.

© SAP 2008 / SAP NetWeaver CE 7.11 EhP1 / Exploring Web Dynpro Java: Goals, Concepts, Benefits / Page 81