Major Feature Changes Between 1.5.x and 2.0

This document discusses some of the major functional changes between Pivot 1.5.x and Pivot 2.0.

Core

- **PIVOT-470** Rename org.apache.pivot.wtkx.WTKXSerializer to org.apache.pivot.beans.BXMLSerializer
  Despite its name and inclusion in the WTK library, WTKXSerializer never had any actual dependency on WTK. In Pivot 2.0, this class has been renamed to BXMLSerializer and moved to the Core library to reflect its more general nature. It has also received a number of updates and improvements, discussed below.

- **PIVOT-514** Create an annotation to specify the default "child" property of a component
  Classes now can specify a default property using the new @DefaultProperty annotation. For example, the org.apache.pivot.wtk.Window class declares a default property of "content", making the <content> element in markup optional. Taking advantage of default properties can significantly reduce the verbosity (and indent depth) of BXML markup.

- **PIVOT-568** Add support for dynamic data binding
  Pivot 1.5 and earlier supported data binding via a load/store model that maps well to client/server applications such as REST clients. However, a more dynamic model, where a property of a target element is automatically updated whenever a source value changes, is also useful in many circumstances.

  Pivot 2.0 adds support for declaratively creating such dynamic binding relationships as well as the ability to define them in code. For example, the following markup creates a binding association between a page variable named "myText" and the "text" property of a Label instance. Any time the value of "myText" changes, the value of the label's "text" property is also updated:

  ```xml
  <Label text="${myText}"/>
  ```

- **PIVOT-546** Bindable improvements
  The Bindable interface allows a caller to easily associate Java code with markup defined in BXML. It defines a single method, initialize(), that is called on the root element of a BXML document once the document has been completely loaded.

  Though Bindable was actually introduced in Pivot 1.5, it did not provide all of the information an implementing class might need. Pivot 2.0 adds arguments containing the serializer's namespace, resources, and location, to the initialize() method. Any BXML annotations defined on the Bindable class are also processed prior to the call to initialize(). This allows the implementing class to get access to the document's namespace (i.e. page variables), the resources that were used to load it, and the location it was loaded from, to perform any necessary post-processing (for example, registering event listeners).

- **PIVOT-537** Move message processing functionality to pivot-core
  Pivot 1.5 included a pub/sub messaging API as part of the org.apache.pivot.wtk.ApplicationContext class. However, like WTKXSerializer, this API didn't actually have a dependency on WTK. It has been moved to org.apache.pivot.util.MessageBus to reflect this.

- **PIVOT-620** Allow serializers to fire events as data is read
  The org.apache.pivot.json.JSONSerializer, org.apache.pivot.serialization.CSVSerializer, and org.apache.pivot.xml.XMLSerializer classes allow an application to read structured content from an input stream. In Pivot 1.5 and earlier, callers had to wait until the Serializer#readObject() method returned in order to access the deserialized data. Pivot 2.0 allows developers to hook into the serialization process and be notified as data is read. For example, JSONSerializer now fires events whenever a string, number, boolean, list, or map is read, and XMLSerializer fires events as elements and text nodes are read. This allows a caller to update the UI incrementally rather than waiting until the entire stream is read, making an application seem considerably more responsive.

WTK

- **PIVOT-553** Add support for named styles
  While Pivot has always provided support for modifying the appearance and behavior of components via CSS-like styles, it did not include support for "named styles" (aka "style classes"). Pivot 2.0 adds this capability, including support for both typed and untyped style selectors.

- **PIVOT-650** Add platform support for SVG images
  Pivot 1.5 and earlier included a set of classes for declaratively constructing drawings using a "shape DOM" or "scene graph", which mirrored the equivalent drawing primitives in the java.awt.geom package. While these classes provided a convenient way to implement retained mode drawing, they were of little practical use, since such drawings had to be constructed by hand - no support was provided for reading standard image formats such as SVG or Adobe Illustrator documents.

  In Pivot 2.0, the "shape DOM" classes have been eliminated and replaced with a new SVG-based Drawing class. This allows developers to freely interchange bitmap-based images (such as JPEG, PNG, or GIF) and scalable vector images in a Pivot application. SVG support in Pivot 2.0 is provided by the SVG Salamander project.

- **PIVOT-555** Re-implement TextArea
  The TextArea component was completely overhauled in Pivot 2.0. It now supports common text editing features including word navigation and undo/redo, and cut/paste behavior has been improved. Word navigation and undo/redo has also been added to TextInput (PIVOT-372 and PIVOT-639).

- **PIVOT-31** Add rich text support to TextPane (formerly TextArea)
  The previous TextArea component has been renamed to TextPane and has received improvements that allow it to be used to present and author rich text. It can also now host sub-components and can be used as a generic text-based layout container. However, it has not been fully tested and is considered experimental in Pivot 2.0.
• **PIVOT-458 Add a "repeatable" property to ListButton**
   A "repeatable" flag has been added to `ListButton` to support "split button"-type behavior. When enabled, a mouse click on the trigger area of the button will show the drop down, but a click on the content area will not and will instead simply "press" the button.

• **PIVOT-654 Simplify editor APIs**
   The editor APIs for `ListView`, `TableView`, and `TreeView` were significantly simplified for Pivot 2.0, making it much easier to implement custom editors in a Pivot application.

• **PIVOT-569 Make ListView selectedItem, etc. notifying properties**
   In previous versions, changes to `ListView`'s "selectedItem" property, `TableView`'s "selectedRow" property, and `TreeView`'s "selectedNode" property did not fire events. This was by design, since the getters for these properties are simply convenience methods. However, the lack of event notification made it impossible to use the new dynamic data binding feature to bind to these properties. As a result, events are now fired when these values change.

• **PIVOT-548 Fire selection change events when selection changes indirectly**
   In previous versions, selection change events in data-driven components such as `ListView` were not fired when a component's selection state changed indirectly (i.e. as a result of a model change, rather than a direct call to a selection setter method). It was the caller's responsibility to listen for both model and selection change events.

   In Pivot 2.0, selection change events are fired both when the selection changes directly as well as when it is changed indirectly.

• **PIVOT-74 Add a "closeable" property to TabPane**
   The `TabPane` component now supports user-closeable tabs.

• **PIVOT-512 Fire tooltipTriggered() event from Component**
   In prior Pivot versions, it was difficult to create tooltips containing custom content. The `Component` class now fires a `tooltipTriggered()` event that applications can hook into to show a custom tooltip. The tooltip delay is now also configurable (PIVOT-563).

• **PIVOT-418 Multiple host windows**
   Though Pivot provides a rich set of window types, they are all internal to the host frame (or applet). A common request from developers was for improved support for multiple native frames. Such support has been added to Pivot 2.0: applications launched via `DesktopApplicationContext` can now easily create multiple top-level host frames, providing a much more seamless desktop experience.

• **PIVOT-579 Provide additional Color schemes**
   Though Pivot has always provided the ability to create custom color schemes, previous versions have included only a single default scheme. Pivot 2.0 includes a collection of color schemes which are optimized for a variety of popular desktop environments.

• **PIVOT-239 Create an Eclipse launcher for org.apache.pivot.wtk.Application and ScriptApplication**
   Pivot now includes an Eclipse plugin to help simplify the task of creating launch configurations for Pivot applications. When the plugin is installed, any class that implements the `Application` interface can be automatically launched via a right-click context menu. Additionally, any BXML file that has a root element that is a subclass of `Window` can also be launched via right-click using the `ScriptApplication` launcher class included in the platform.

Complete release notes are available [here](#).