TuneIn set up.

for version 1.0.0

This is for adding dates for date coded banner content, videos, or just the changing of text for tune in's. The video player components access this directly when you set the useTuneIn in value to true.

IMPORTS:

In the fla you only need to import 1 class, nothing gets added to the library.

```
import com.blt.components.TuneIn;
```

PUBLIC PROPERTIES:

These are read only values, you can't set them.

isReady:Boolean = false
Is true after parseDate is called, used in the video player.

currentTimeZoneOffset:Number
This returns the users time zone offset, that is the distance in time between the users time zone and the UTC / GMT zone. The returned value is in hours.

currentframe:uint
Returns the frame value for the current date that was passed in the addDate method.

currentId:String
Returns the date id value for the current date that was passed in the addDate method.

currentVideoId:String
Returns the date id value for the current date that was passed in the addDate method, that doesn't have useWithVideo property set to false.

PUBLIC METHODS:

addDefault(_id:String = "", _frame:uint = 0, _swfId:String = null):void
This method is used to add videos to the videoListManager.

- **_frame:uint ::** This is a integer, it was added to allow the user to pass frames by number, it would work like this.
  var frameLabel:uint = TuneIn.currentframe;
  or if using a swfId
  var frameLabel:uint = TuneIn.getCurrentframeBySwfID("swf_id");
  gotoAndStop(frameLabel);

- **_id:String ::** This is a string, this is the value that the video player will use to grab date coded videos provided useTunein is set to true on the video player. This value can also be used to target a frame label.
  var frameLabel:String = TuneIn.currentId;
  or if using a swfId
  var frameLabel:String = TuneIn.getCurrentIdBySwfID("swf_id");
  gotoAndStop(frameLabel);

addDate(_date:Object, _id:String = "", _frame:uint = 0, _useWithVideo:Boolean = true, _swfId:String = null):void
addDate is where you will add the individual dates, along with the corresponding frame and id value.

- **_date:Object ::** Is an object that contains the year, month, day, and time if needed. Months are now 1 based, so you can enter 1 for january, not 0.
  TuneIn.addDate(**{ year:2014, month:4, day:8, hours:14, minutes:30 }**, "date3", 1, false, "default");

- **_id:String ::** This is the value that the video player will use to grab date coded videos provided useTunein is set to true on the video player. This value can also be used to target a frame label.
  TuneIn.addDate({ year:2014, month:4, day:8, hours:14, minutes:30 }, **"date3"**, 1, false, "default");
  var frameLabel:String = TuneIn.currentId;
  or if using a swfId
  var frameLabel:String = TuneIn.getCurrentIdBySwfID("swf_id");
  gotoAndStop(frameLabel);

- **_frame:uint ::** This is a integer, it was added to allow the user to pass frames by number, it works like this.
  TuneIn.addDate({ year:2014, month:4, day:8, hours:14, minutes:30 }, "date3", **1**, false, "default");
  var frameLabel:uint = TuneIn.currentframe;
  or if using a swfId
  var frameLabel:uint = TuneIn.getCurrentframeBySwfID("swf_id");
  gotoAndStop(frameLabel);

- **_useWithVideo:Boolean ::** If your using date coded videos but not all the dates you added have video associated with them, you can pass false for this value, it will tell the tuneIn not to send this value when the video player asks for the current video id, it will send the previous date that doesn't have this set to false.
  TuneIn.addDate({ year:2014, month:4, day:8, hours:14, minutes:30 }, "date3", 1, **false**);

- **_swfId:String ::** The swfId only needs to be used in very special cases. If you have an expanding component that has different dates in the component swfs then you need to pass a swf id, this will group all the values under that string. You have to pass the swf id value when getting your frame or date id, otherwise you will get the default values.
  TuneIn.addDate({ year:2014, month:4, day:8, hours:14, minutes:30 }, "date3", 1, false, **"default"**);

addDateRange(_startDate:Object, _endDate:Object, _day:uint, _id:String = "", _frame:uint = 0, _useWithVideo:Boolean = false, _swfId:String = null):void
addDateRange allows you to add a start and end date which is passed using the same object values as the date for addDate. You pass a day value, use the ones listed in the tune in class file, for friday it would be TuneIn.FRIDAY. Whenever that day shows up within the start and end dates it will return this id or frame value.

- **_startDate:Object ::** Is an object that contains the year, month, day, and time if needed.
  Months are now 1 based, so you can enter 1 for january, not 0.
  TuneIn.addDateRange(**{ year:2014, month:4, day:8 }**, { year:2014, month:6, day:15 }, TuneIn.FRIDAY, "fridays", 1, true, "user");

- **_endDate:Object ::** Is an object that contains the year, month, day, and time if needed.
  Months are now 1 based, so you can enter 1 for january, not 0.
  TuneIn.addDateRange({ year:2014, month:4, day:8 }, **{ year:2014, month:6, day:15 }**, TuneIn.FRIDAY, "fridays", 1, true, "user");

- **_id:String ::** This is the value that the video player will use to grab date coded videos provided useTunein is set to true on the video player. This value can also be used to target a frame label.
  TuneIn.addDateRange({ year:2014, month:4, day:8 }, { year:2014, month:6, day:15 }, TuneIn.FRIDAY, **"fridays"**, 1, true, "user");
  var frameLabel:String = TuneIn.currentId;
  or if using a swfId
  var frameLabel:String = TuneIn.getCurrentIdBySwfID("swf_id");
  gotoAndStop(frameLabel);

- **_frame:uint ::** This is a integer, it was added to allow the user to pass frames by number, it would work like this.
  TuneIn.addDateRange({ year:2014, month:4, day:8 }, { year:2014, month:6, day:15 }, TuneIn.FRIDAY, "fridays", **1**, true, "user");
  var frameLabel:uint = TuneIn.currentframe;
  or if using a swfId
  var frameLabel:uint = TuneIn.getCurrentframeBySwfID("swf_id");
  gotoAndStop(frameLabel);

- **_useWithVideo:Boolean ::** If your using date coded videos but not all the dates you added have video associated with them, you can pass false for this value, it will tell the tuneIn not to send this value when the video player asks for the current video id, it will send the previous date that doesn't have this set to false.
  TuneIn.addDateRange({ year:2014, month:4, day:8 }, { year:2014, month:6, day:15 }, TuneIn.FRIDAY, "fridays", 1, **true**, "user");

- **_swfId:String ::** The swfId only needs to be used in very special cases. If you have an expanding component that has different dates in the component swfs then you need to pass a swf id, this will group all the values under that string. You have to pass the swf id value when getting your frame or date id, otherwise you will get the default values.
  TuneIn.addDateRange({ year:2014, month:4, day:8 }, { year:2014, month:6, day:15 }, TuneIn.FRIDAY, "fridays", 1, true, **"user"**);

setTestDateByDate(_date:Object):void
This method allows the user to set a test day for local testing in the flash IDE only, once it's put up on the server the test day will be ignored. This is instead of changing your machine date for testing.
TuneIn.setTestDateByDate({ year:2014, month:4, day:8, hours:14, minutes:30 });

setTimezone(zone:String):void
This is for special cases where you want an event to happen regardless of time zones, so if the event starts at 9pm in new york you want the change to happen at 6pm in los angeles. You add your dates as usual but set a timezone that the dates and times are local to. TuneIn has a list of time zone values that you can pull directly from the class. For eastern it would look like **TuneIn.setTimezone(TuneIn.TIMEZONE_EASTERN);**

parseDate(_swfId:String = null):void
Call this after all the dates have been added including the test date and timezone has been set if being used. When this is called it organizes all the dates and sets the current frame and id based on the users machine date or test date if set. If using swf id's you need to pass it here.

BASIC TUNE IN SET UP:

```
import com.blt.components.TuneIn;

TuneIn.setTimezone( TuneIn.TIMEZONE_HAWAII );

TuneIn.addDefault( "default" ); // default date

TuneIn.addDate( { year:2014, month:4, day:1 }, "date1" );
TuneIn.addDate( { year:2014, month:4, day:5 }, "date2" );
TuneIn.addDate( { year:2014, month:4, day:8,  hours:14, minutes:30 }, "date3" );
TuneIn.addDate( { year:2014, month:4, day:25, hours:12, minutes:50 }, "date4" );

TuneIn.setTestDateByDate( { year:2014, month:4, day:25, hours:12, minutes:50 } );
TuneIn.parseDate();

trace( "TUNIN TEST: " + TuneIn.currentId );
```

TUNE IN SET UP USING SWF ID's AND DATE RANGE:

```
import com.blt.components.TuneIn;

TuneIn.addDefault( "default", "user" ); // default date

TuneIn.addDate( { year:2014, month:4, day:1 }, "date1", "user" );
TuneIn.addDate( { year:2014, month:4, day:5 }, "date2", "user" );
TuneIn.addDate( { year:2014, month:4, day:8 }, "date3", "user" );
TuneIn.addDate( { year:2014, month:4, day:25 }, "date4", "user" );
TuneIn.addDateRange( { year:2014, month:4, day:25 }, { year:2014, month:6, day:15 }, TuneIn.FRIDAY, "fridays", 1, true, "user" );

TuneIn.parseDate( "user" );

trace( "TUNIN TEST: " + TuneIn.getCurrentIdBySwfID( "user" ));
```