agileWORKFLOW Manual

Contents

1. agileWORKFLOW

This documentation explains the concepts and the scheduler in details. For the installation instructions please have a look at the agileWORKFLOW Administration documentation.

2. The repository

Definition

The Repository holds all the information needed by the scheduler: the object definitions and the execution history. It is a collection of tables and views and requires a supported Database in order to work.

For a list of supported RDBMS please look at the System Requirements .

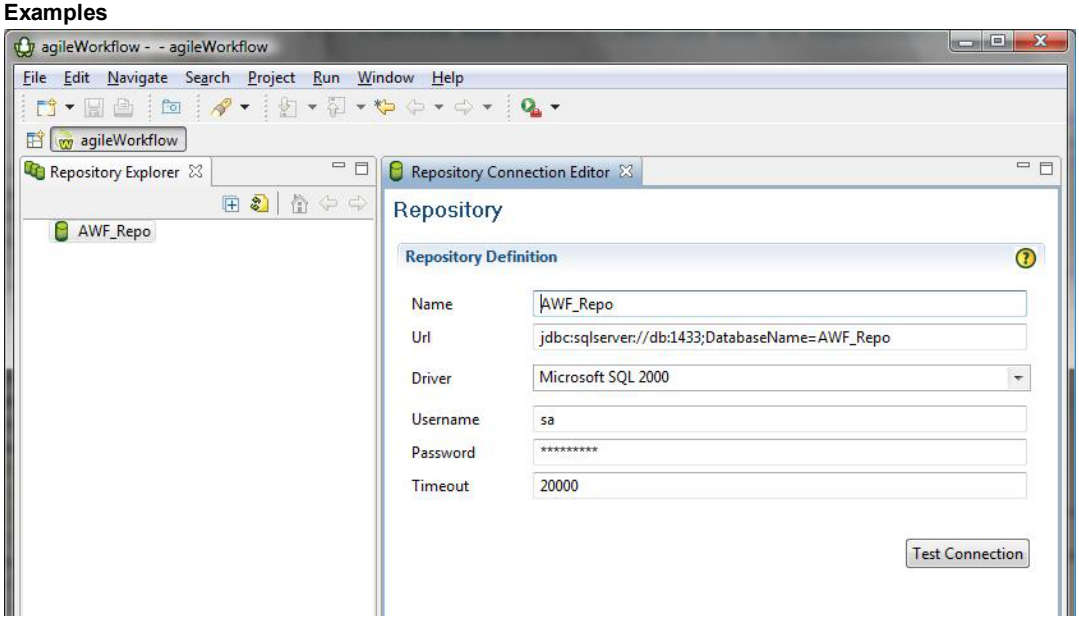
Every application part of agileWORKFLOW needs access to a repository. Repository information is stored in the *config/connections.xml* file.

The awf_ui graphical console allows one to create and update object definition objects as well as monitor the execution info status. The awf_agents retrieve information from the repository, execute the workload and update the repository with the results.

Properties

- Name: The name of the repository. This value is left entirely at the discretion of the administrator, it is recommended to give a name to each repository that is representative of its contents (or to the specific systems its meant to
- URL: The JDBC URL used to connect to the database containing the repository. The format of this setting varies for each Database.
  - IBM DB2: **URL :** jdbc:db2://*[Host name]*: *[Port Number]*/*[Database Name]*
  - Microsoft SQL Server 2000 / 2005: **URL :** jdbc:sqlserver://*[Host Name]*:*[Port Number (default: 1433)]*;DatabaseName= *[Database_Name]*
  - Oracle 9i: **URL :** jdbc:oracle:thin:@//*[Host Name]*:*[Port Number (default: 1521)]*/*[Service Name}*
- Driver: The list of supported JDBC driver is found here, simply select the driver required to connect to your chosen rdbms.
- Username: The username used to connect to the database.
- Password: The password associated with the specified username.
- Timeout: The amount of milliseconds to attempt to connect to the database before failing.

Examples



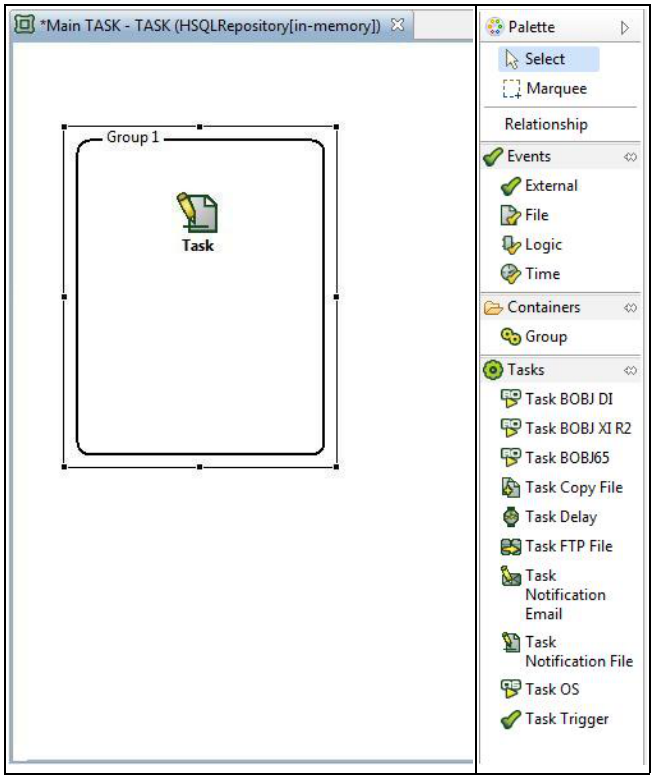
3. Description

The Diagram is the main canvas of your workflow. There are two types of digrams: Agent and Task. The Agent Diagram displays agent/cluster information and the Task Diagram displays the workflow objects such as events, tasks and groups. The Task Diagram editor allows one to drap and drop the workflow objects as well as create relationships between them. Lastly, the diagram editor displays the latest execution info for the objects contained in the digram.

Modifying a workflow

It is important that no agent be in execution when a workflow is being updated. Therefore, please make sure the agent has been gracefully shutdown before saving any modifications.

Examples



4. Description

An Agent is a process responsible for evaluating events, dispatching groups and executing task jobs. There can be multiple agents active per repository and they are synchronized by a single Dispatcher Service running as part of one agent.

Note however that an agent can only connect to a repository at a time.

Clustering

A Cluster is an abstract container for agents. There is no physical process for this object. Tasks can be linked to a cluster instead of one single agent. During execution time, the dispatcher will select the agent with the lowest load and assign the task job to it. This balances the load on the agents and ensures the execution of that task.

Properties

- Sleep Time (ms): # milliseconds to sleep during a cycle. Default value is 5000 ms.
- UDP Service Port: port # on which the agent listens for communication; must be different on every agent running on the same server

Note that events cannot be linked to a cluster.

Services

An agent contains the following (3) services.

- Dispatcher Service
- Event Service
- Execution Service

Note that only one Dispatcher Service can be active (Started) per repository. Multiple running Event Services or Execution Services are allowed per repository and in fact recommended for non-trivial workflows.

Installation

Please take a look at the Agent section in the Administration Document for installation instructions.

Monitoring

A few actions are supported by the graphical console (awf_ui). They can be found under 'Action' either in the Repository Explorer View, the Agent Diagram or the Agent Monitoring View.

- Stop Agent: Sends request to the agent to gracefully shutdown
- Reset Agent: Forces agent status to Stopped in the repository. Use this option only when an agent was not shutdown cleanly.
- Start Event/Dispatcher/Execution Service: Starts the service. *
- Stop Event/Dispatcher/Execution Service: Stop the service. *

* Only present in the Agent Monitoring View

The Agent Monitoring View displays workload information such as

- number of used tasks execution slots
- number of defined tasks execution slots
- history log for the agent and the services

4.1. Dispatcher Service

Description

The Dispatcher Service is the 'engine' of the workflow. It initializes the objects, determines what to run next, processes forced executions, recovers from dead agents and balances the workload in a cluster.

Note that only one dispatcher service can be started by repository; enabling any other will result in an error.

Dispatching rules

Definitions

- Triggering event
  - The status is True.
  - The current instance did not trigger the same execution unit before.
  - The current instance is not expired (time events only).
- Bypassing
  - The status is False.
  - The current instance did not trigger the same execution unit before.
  - The current instance is not expired (time events only).
- Execution Unit: either a sub-group or a task.
- Expired Instance
  - Event definition is a time event
  - The time between the moment the event was triggered and the moment this same event is being handled by its associated agent must be within the limit set in the 'expiration' property of the event, otherwize this instance is discarded.

    The default 'expiration' time of '-1' results in a timeout of 6 hours for the following time based events: Once, Minute, Hourly, Daily; and 12 hours of timeout for Monthlty and Yearly events.
- Elected Descendant
  - Single Forced Execution:
    - Task: only the task is executed; there are no no elected descendants.
    - Group: all contents of the group
  - Cascade Forced Execution:
    - Task: all connected objects
    - Group: all contents of the group are elected descendants as well as all connected objects

Waiting rules

A task will block at 'Waiting' in all these cases:

- The task agent is not running.
- The task agent execution service is not Started.
- The task agent execution service is dead.
- A task needs to lock a resource.
- Task is linked to a cluster and there are no free agents in that cluster.

Normal Execution

When an object is executed, it passes through the following sequence: Initialized to Ready to Waiting to Dispatched and to Running.

- A top group is executed when:
  - It is triggered by a Triggering Event
- An execution units is executed when:
  - It is triggered by a Triggering Event (if one is associated with the Execution Unit).
  - Its parent is Running.
  - It has acquired all its required resources and an agent is available to dispatch it.

Bypassed Execution

When an object is bypassed:

- All group contents are bypassed. Group events become False.
- An execution unit is bypassed when:
  - It is triggered by a Bypassing Event.
  - Its parent is Running.

Forced Execution

A forced execution applied to an execution unit will force all the parents of the is execution unit to Running. All the descendants of the cascaded forced execution will be executed under normal dispatching rules.

- A top kroup will be executed under any circumstances.
- An execution unit is executed when:
  - Its parent is Running.
  - It has acquired all its required resources.

Rerun

During a workflow run, some tasks might fail, leaving other execution units in a bypassed status. The rerun options allows the administrator to only re-evaluate the failed tasks and the ones that were bypassed.

Recovery

If the flow is stopped for any reason (Server crash, Agent not responding, etc) the following rules are applied upon recovery:

- Execution units with status Ready will become Recovered; re-execution will occur next.
- Execution units with status Running will switch to the final status Aborted.

Properties

- Sleep Time (ms): # milliseconds to sleep during a cycle. Default value is 10000 ms.
- Enable (boolean): Start the service when the agent starts.

4.2. Event Service

Description

This service is also part of the agent and its purpose is to evaluate all events besides External which have been associated for a given agent.

The service wakes up after its sleep time, looks for events to process, evaluates them and updates their status in the database.

Properties

- Sleep Time (ms): # milliseconds to sleep during a cycle. Default value is 10000ms.
- Enable (boolean): Start the service when the agent starts.

4.3. Execution Service

Description

This service is also part of the agent. This service looks for all dispatched Tasks that are assigned to its agent, creates jobs for these tasks, executes them and updates the final result in the repository.

Properties

- Sleep Time (ms): # milliseconds to sleep during a cycle. Default value is 10000ms.
- Enable (boolean): Start the service when the agent starts.

- Max Threads (integer): # of jobs to run in parallel. Default is 5.

5. Variables

Description

Variables act as placeholders/aliases and are found in variable enabled fields for either events or tasks. Its usage eliminates the hardcoding of file paths, server names, configuration parameters, names, etc. Furthermore, dynamic variables are much more flexible since they can query database tables, connect to other servers, etc.

A variable is expanded only on-demand, when a user wishes to view manually expand or when a service executes an event or a task. The variable value is always assumed to be a string.

Static vs Dynamic

Static variables are aliases. For a given key there can be only one value. They can be used for file/directory names, server names, etc.

A dynamic variable is a wrapper on top of javascript code. The Mozilla Rhino engine also supports Java objects to be used with Javascript. These variables can perform SQL queries, date calculations, etc. They are completely user defined.

Usage

Some editor text fields allows for the use of variables. These will have a small expand button to their left.

Example of a variable text field

Assume that "test" is a static or dynamic variable stored in the repository. To use "test" in the context of a variable field, simply surround it in "{" and "}".

So a variable field might look like this: "{ test }" or "{test}". The number of spaces after the "{" or before the "}" is ignored.

However, if the entry contains spaces such as "test 123", the variable entry must preserve that space. I.e.: "{test 123}" will work but "{test 123}" will not.

If a variable text field contains "{ test}.txt" but the "test" entry does not exist, then the result will be the actual string "{ test}.txt".

Limitations

The static variable name is limited to 254 characters.

The static variable value is limited to 2048 characters.

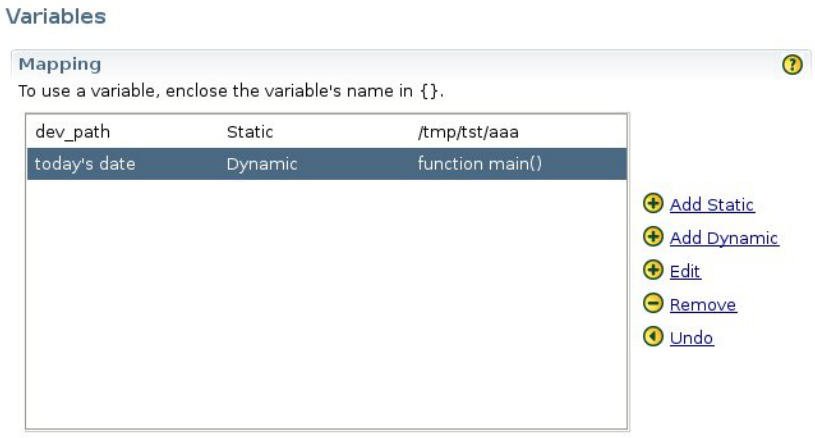
A dynamic variable might cause runtime exceptions or be caught in infinite loops. A user must keep this in mind. Any exceptions will result in task and event failures with the error message stored in the instance log.

The dynamic variable code is limited to 2048 characters.

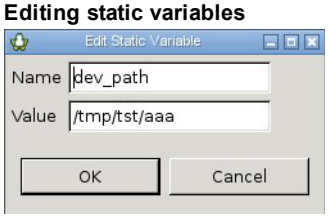
Nesting variables such as "{test {test2} }" is not supported. In this case, only "{test2}" would get expanded.

Examples

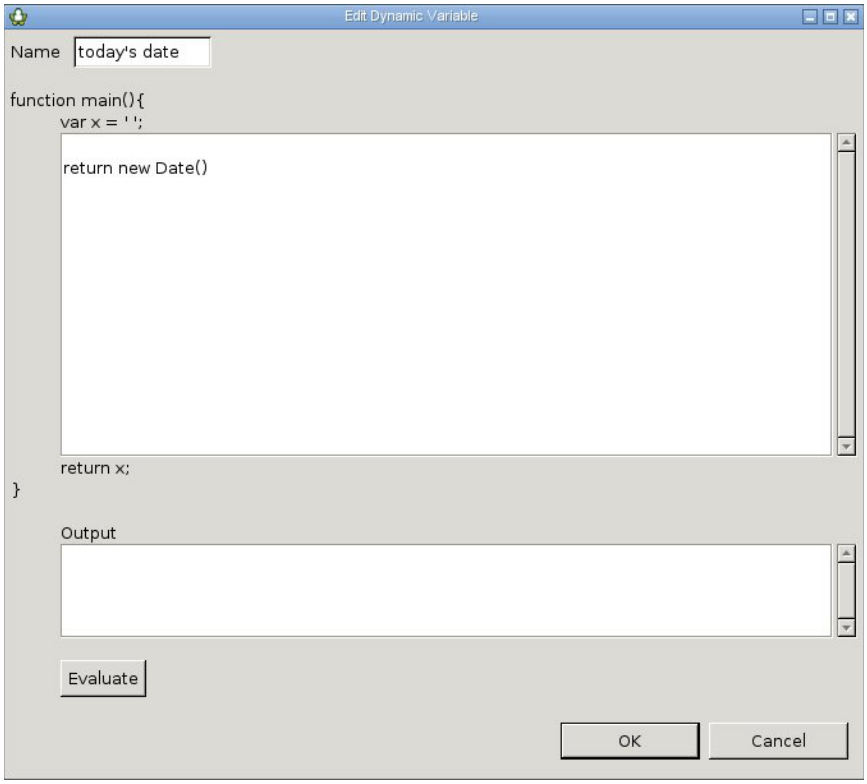
Creating and modifying variables



Editing static variables



Editing dynamic variables



Please look at the Rhino Javascript Documentation for scripting Java in JavaScript.

6. Instances and Execution History

Description

An instance represents the execution of a task or an event. All the information for that particular execution is associated with a single instance. If you evaluate an event ten times, then you will have ten separate instances. Each instance keeps property and errors logs which can be viewed later on. The evaluation time is also stored.

The object instances can be tracked for each object using the Instance Explorer.

The execution history context represent all the instances for all objects in the repository.

Properties

The following information is stored per instance:

- The object definition which was evaluated
- The start and end date
- The complete history of transitions
- The executing agent

Each history transition has:

- The status i.e. Initialized, Ready, True, etc
- The timestamp when it happened
- The time spend in that status

The Instance explorer can display how much time an instance spend in a particular status, the total time spent, the agent used, as well as which instance was used to trigger the object, etc.

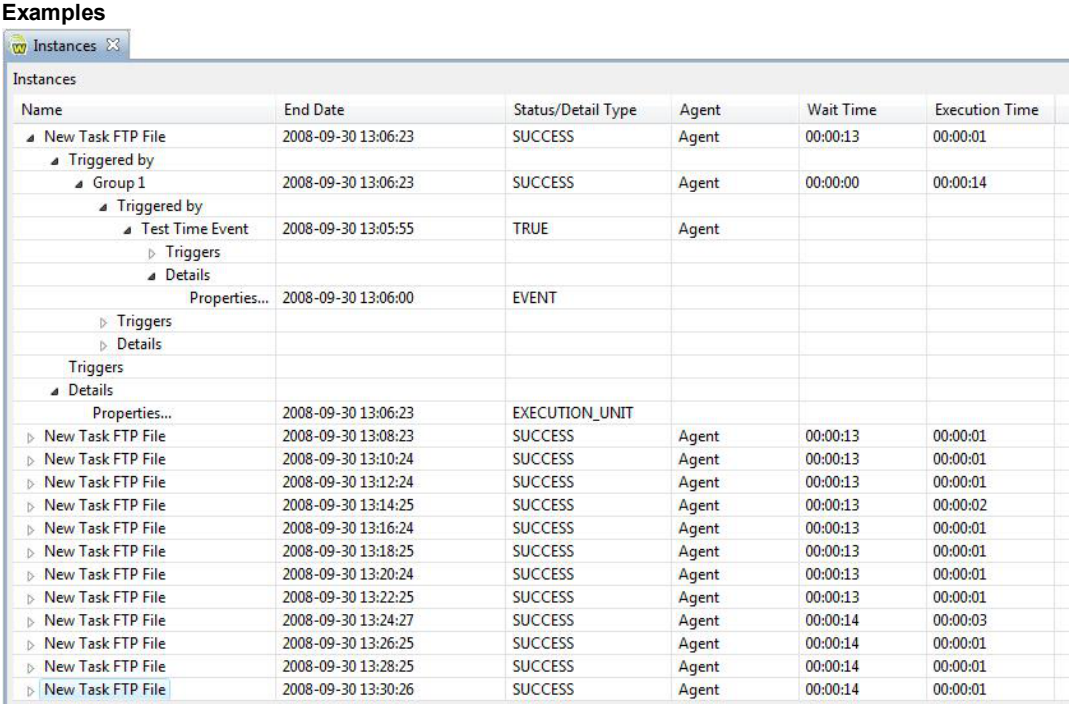
The waiting time is the time between the moment an object instance status was set to Initialized (Ready to be taken in charge by an Agent) and the actual time at which it was set to Running.

The execution time is the time between the moment an object instance status was set to Running and the actual time at which it was set to a final status (Success, Failure or Aborted).

History flow

INITIALIZED	â	READY	â	WAITING	â	DISPATCHED	â	RUNNING	â	SUCCESS
		RECOVERED	Â	RECOVERED	Â	RECOVERED	Â	RECOVERED	Â	FAILURE
		BYPASSED		BYPASSED		BYPASSED		BYPASSED		ABORTED

Examples



7. Events

Description

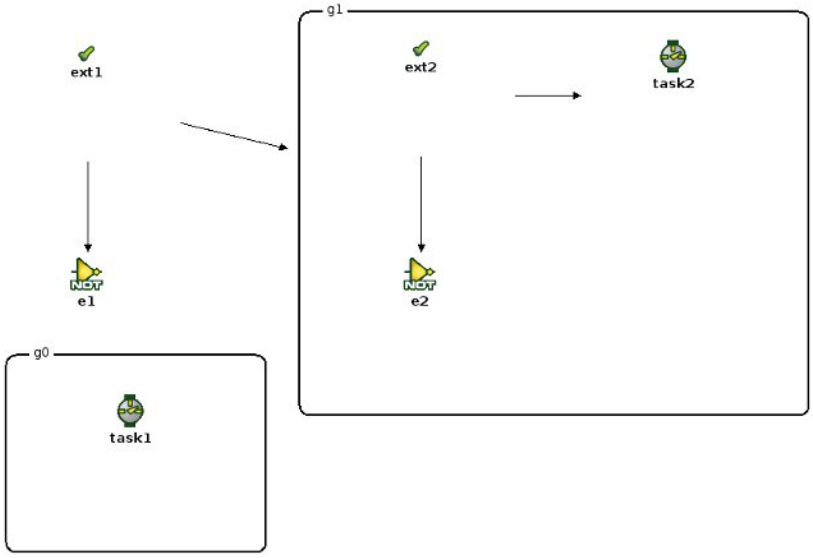
The purpose of an event is to evaluate a condition and get a boolean output in return (True/False).

Triggering and Participation rules

An event can only trigger or participate in logic events at its own level only, therefore triggered groups, tasks or event logics must have the same parent group as the event.

Example

In this example, "ext1" could not trigger "task2" and could not participate in "e2". Furthermore, "ext2" could not trigger "g0" nor "task1" and could not participate in "e1".



Event evaluation

Event evaluation will differ for events with a group and those without.

A group event is evaluated once per every time its parent group is running (Running status). When the event finishes with a final status, it waits until its group is running again.

An event without a group is continously being polled at a defined interval (see the agent's serviceEvent for the sleepTime interval). This type of event is the only one that can trigger top groups.

7.1. External Event

Description

The external event is an event whose status can be changed at the command line by an External tool (furnished with AWF) or via the graphical console (awf_ui).

Changing the status to TRUE from any other status will result in the generation of a new instance of the external event, also triggering the associated Execution Units (although passing from TRUE to TRUE will not result in the launching of the Execution Units; in such a case, a FORCED TRUE status must be set).

This event can also be included inside a Group, however it can only be triggered from the outside if it has been Initialized first.

Examples

7.2. File Event

Description

A file event checks for the presence of a file. If the file is present, then the event is true. Otherwise the event is false. A new file event instance will be generated when its status changes from true to false or false to true.

Properties

- Filename (string): path of the file to check
- Delete (boolean): delete the file after triggering

Examples

7.3. Logic Event

Description

A Logic Event is the result of a Logic operation applied on the multiple possible input Event Statuses.

These statuses will be evaluated and the logic calculation applied to the inputs every time a new instance if a triggering event is applied to the logic door. The result of this logic operation becoming the new status of the Logic Event. The resulting status of a Logic Door depends upon the type of the door and the resulting shortcuts it allows for.

'And', 'Not' and 'Or' operations are supported, with 'And' and 'Or' operations using shortcuts. A 'And' logic door will output a new instance with a status of FALSE as soon as one of its entries is FALSE , whereas a 'Or' Logic Door will output a new instance with a TRUE status as soon as one of its entries if TRUE.

Properties

- Participants: list of inputs (other events only)
- Type (And, Or or Not)

Examples

7.4. Time Event

Description

A Time event is used to schedule execution units. Its status will remain true across its entire execution, but new instances of the Time Event will be created each time the event 'occurs'. As with every other event type, associated Execution Units will be triggered when a new instance of the Time Event is generated.

The Time Event has a unique way of being evaluated. It's the only event that possesses a ' Event Start Date'. When the Agent associated to a Time Event is run for the first time, an instance of the Time Event is created for each time it should have been called since its ' Event Start Date ' up to now. These instances will not have been executed, so no Agent will be associated to it.

Expiration

Time events are associated an Expiration Time , which causes these event to receive the EXPIRED status if they are not executed within their Expiration period. The Default Expiration Time for monthly and daily events is 12 hours , 6 hours for all other event types. The default expiration time is denoted by a ' -1 ' in the Expiration field.

Properties

- Start Date: date before which the event does not react
- End Date: date after which the event does not react
- Frequency: the frequency of the triggering (once, minute, ... yearly)
- Calendar: the caledar to use; the default is the base calendar
- Expiration time in seconds; -1 means using the default value

Examples

7.4.1. Calendars

Description

The purpose of the calendar is to exclude dates from the scheduling. Time events associated with these calendars will ignore the dates in the excluded dates. They can be used to avoid scheduling during holidays or during downtimes.

The base calendar is the default calendar used by the workflow. Any excluded dates in this calendar will have an impact on every time event that uses the default calendar.

Calendar creation

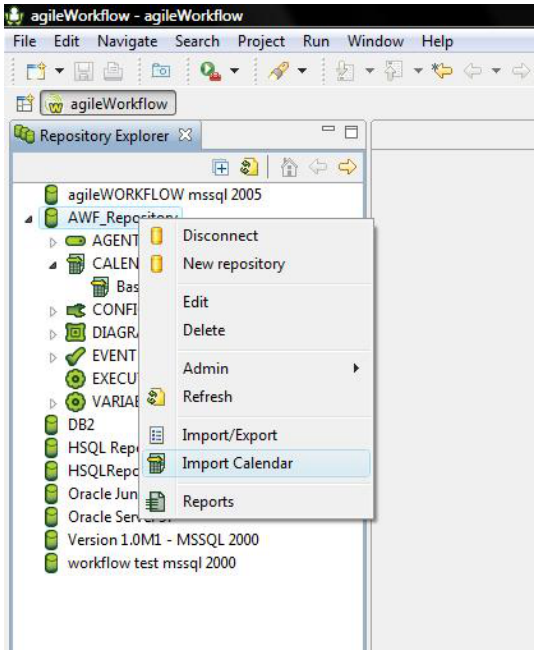
In order to create a new calendar, you need to create a text file and add excluded dates in the YYYY-MM-DD format as in:

- 2007-01-29
- 2007-01-30
- 2007-01-31
- 2007-02-01
- 2007-02-02

After saving the file, connect to a repository, right-click on the repository name and select 'Import Calendar' option. The wizard will prompt to you either create a new calendar or to update an existing one. When updating an existing calendar, you can either append new dates or overwrite the whole exclusion list.

Examples

Importing a calendar



7.5. Generated Events

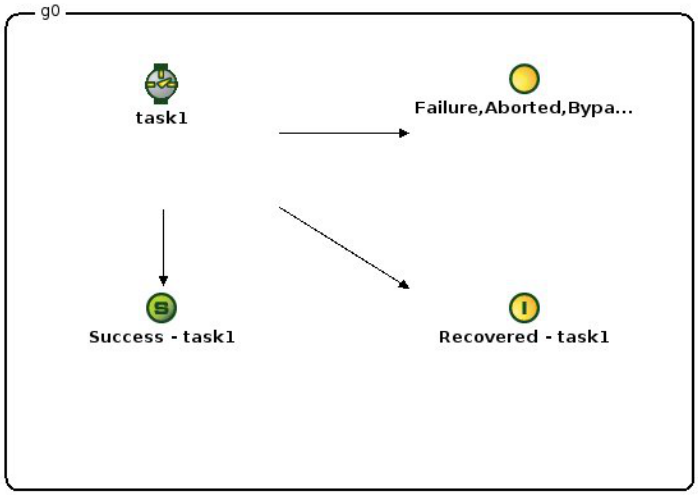
Description

A generated event contains a list of target statuses. If the task or the group passed to one of the target statuses, the event evaluates to true; otherwise it will fail.

The target statuses can be the combination of any possible statuses.

Note however that once the generated event becomes true, it will not change its status.

Examples



Suppose in this example "task1" fails (Failure status). The "Success - task1" event will evaluate to false; however the "Failure, Aborted, Bypassed - task1" will evaluate to true.

7.6. User-Defined Event

Description

A User-Defined Event contains Javascript code which is evaluated by an agent's ServiceEvent. Moreover, a user fills the 'Main' function and returns a value which will either evaluate to True or False.

Any exceptions encountered will result in a False state.

This event should ALWAYS be used within a group since it is evaluated on each ServiceEvent cycle.

Properties

- Javascript code: the javascript code that will be executed by the event. Must return a boolean value.

Examples

7.7. SQL Event

Description

The SQL event connects to a database using a SQL configuration, sends a query defined in its properties and evaluates the result. By default, the javascript code will compare the first column/first row with 0 and return the result. However the code can be completely customized to perform different comparisons.

If one customizes the code, one should take note at the following variables which are injected by default for a SQL event:

- username: the username defined in the SQL configuration
- password: the password defined in the SQL configuration
- driverClass: the class of the driver defined in the SQL configuration
- url: the URL from the SQL configuration
- query: the SQL query defined in the event

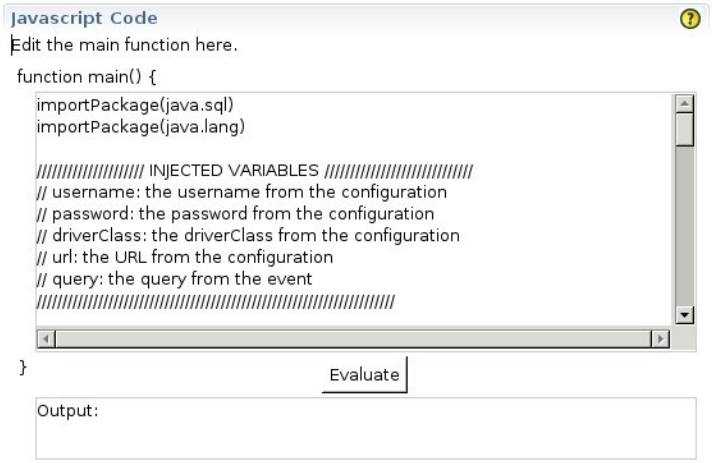
This event should ALWAYS be used within a group since it is evaluated on each ServiceEvent cycle.

Properties

- Configuration: SQL configuration
- SQL query: the select statement

Examples

The event code section:



The event query section:

8. Execution Units

Description

An execution unit is either a task or a group.

Any execution unit can be triggered by a single event. Once they are triggered, they get evaluated by an agent and an instance is created to represent that evaluation.

8.1. Groups

Description

The group's role in the workflow is to contain events and other execution units (groups or tasks). Once a group is triggered by an event, the workflow dispatcher initializes all the group's contents and starts dispatching to the agents.

A group is running until all of its contents are in a final status.

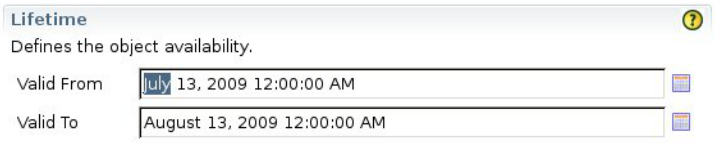
For a group to finish with the success status, all of its execution units must finish with either the success, the bypassed or the cancelled status. Otherwise, the group will finish with the failure status.

Lifetime

Groups can have a lifetime (from / to) which represent the scheduling availability. If the evaluation date is before the "from" date (if any) or after the "to" date (if any) then the group is considered to be invalid.

If the lifetime is invalid the group and its contents will be bypassed.

Any of the two fields are optional.



In the example above, if the execution date is not between July 13 2009 and August 13 2009 then group will be invalid.

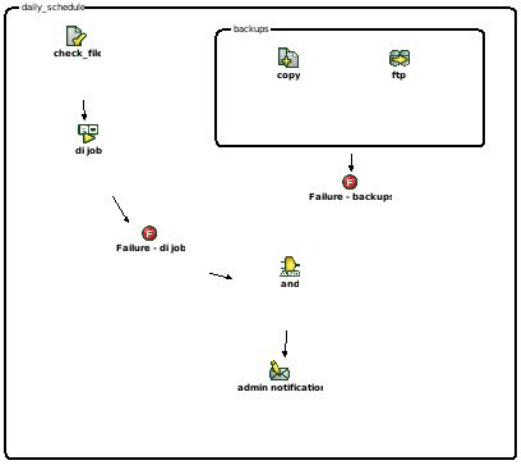
Looping

A group can be iterative. The looping section describes the iterative process in detail.

List and variable mapping inside a group

Examples

A group and its contents



8.1.1. Loops

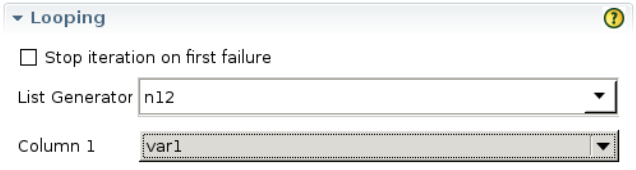
Description

A group can be used to iterate over the contents of a list, one row at a time. The group remains at running until all rows are consummed. The contents of the group are initialized and executed during each iteration.

A looping group

To make an iterative group, a list generator has to be associated to the group. During the association, a list generator is selected and a mapping consisting of the generated list columns to static variables has to be defined. The option "Stop iteration on first failure" will stop a group iterating if any task finishes with a failure or an aborted status. By default, this option is false implying that a group would iterate over all values of the list generator.

List and variable mapping inside a group

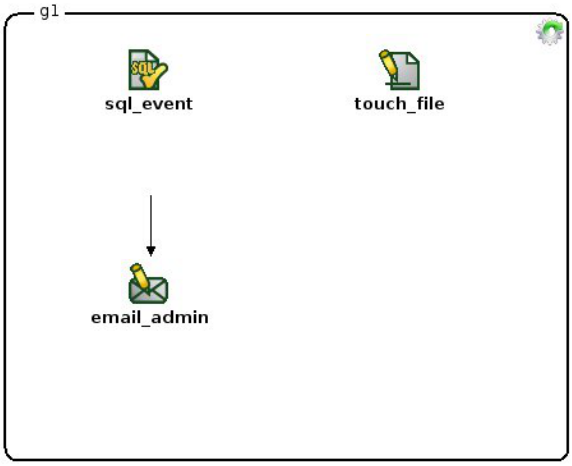


In this example, during each iteration, the static variable "row_number" will be updated to the list's current iteration item. Since static variables are global, everybody including the group's contents are able to use the changed values.

Note however that it is generally a bad idea to map list columns to the same static variables.

Examples

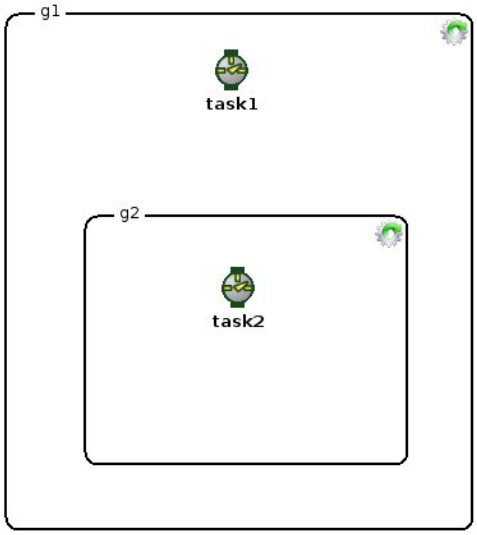
A looping group



Suppose in this example that the group iterates over a list of 5 items ["1", "2", "3", "4", "5"] mapped to a static variable "row_number". Also, touch_file uses "{row_number}.txt" as its filename argument.

During the whole execution, group "g1" remains at running. However, there will be 5 instances of "sql_event", "touch_file" and "email_admin" created assuming "sql_event" always evaluated to true. There will be also 5 files created "1.txt", "2.txt", "3.txt", "4.txt" and "5.txt"

A nested loop



Assume in this example that group "g1" iterates over a list of 5 elements and group "g2" iterates over a list of 3 elements.

At the end of the execution, there will be:

- 1 instance of group "g1"
- 5 instances of task "task1"
- 5 instances of group "g2"
- 15 instances of task "task2"

8.2. Resources

Description

Resources can be assigned to tasks and given fractions of how much they consume it when they are executing. A task requiring 100% will not allow any other tasks to execute (exclusive lock). However, If task T1 is being executed and uses 10% of resource A then task T2 may proceed

with execution only if its resource needs are equal or below 90%.

A task might use as many as resources as possible. If one resource cannot be acquired, it then waits until all resources it needs can be acquired (all or nothing). Also, a resource can be used by every task.

Scenario

Given the following tasks:

- T1 priority 100 resource A 10%
- T2 priority 90 resource A 91%

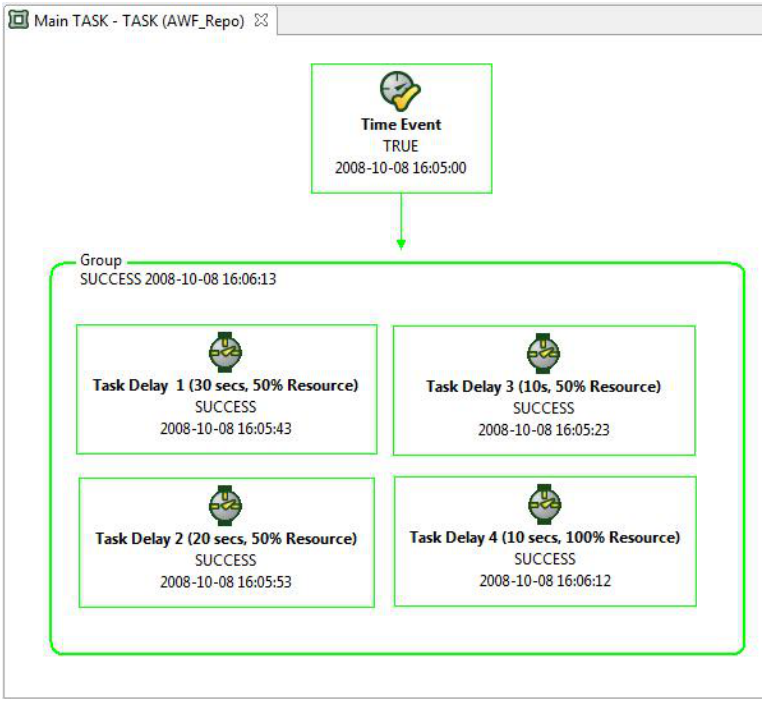
In this example, T1 will start being processed right away because it has the highest priority and locks down resource A for 10%.

During this time, T2 has to wait because it cannot acquire the 91% of resource A it needs.

When T1 is finished, it releases its lock on the resource and then T2 starts processing right away.

Examples

Execution run with resources



8.3. Tasks

Description

A task is an execution unit that executes some work when processed by an agent. A task needs to be part of a group and might need a configuration depending on its type.

Properties

- Priority

  Tasks are dispatched in order of Priority, a property of each task set in the Execution Unit Definition properties page of the task. The default priority of a task is 50, and it can be set between 0 and 99. The Dispatcher service uses the Priority property of tasks to determine the order in which tasks will be dispatched (the higher the priority means the faster the task will be dispatched to a service execution).

- Resources

  Tasks can be assigned resources in order to insure that two different tasks requiring exclusivity to the same resource are not run concurrently (by setting both tasks to 100% of the shared resource) or simply to allow for proper sharing of a limited resource. For example, by setting a number of tasks to reserve 33% of a same resource slot we insure that only 3 of all those tasks can be run

concurrently at any time. Resources are an abstract concept in agileWORKFLOW, they can represent any real or virtual resource. A task requiring to lock a table in a database can take advantage of resources to do so. A resource is created and associated with every task that require access to that table in the database. The task requiring the table to be locked will be set to require 100 % of that resource slot, whereas every other task requiring access to it but not exclusive access can reserve 1% or more of the resource slot in order to insure that they are not run at concurrently.

- Max Duration

  Some tasks can be assigned a Max Duration time (in milliseconds). This value is challenged against the task's execution time by the Execution Service. The Execution Service executes this verification once per Sleep Time Interval, and will attempt to ABORT the task (depending upon the task type).

- Retry Count

  This parameter acts upon the evaluation of a task. If a task evaluates internally to Failure, the task is re-evaluated again for up to the retry count parameter or until a non-Failure status is returned (such as Success).

  By default, the retry count is 0. There is also a pause of 5 seconds between each re-evaluation.

  Please note that these tasks do not currently support Retries:

  - FTP
  - Email
  - File Notification
  - BO 6.5
  - Delay
  - Trigger

8.3.1. Task Delay Definition

Task Delay Definition Tab description.

Task Delay Properties

Contains properties specific for a Task Delay :

- Expected status - the status with which the task execution will finish.
- Time to sleep - the time used by the task to sleep before returning from execution with the desired status.

8.3.2. Task OS Definition

Task OS Definition Tab description.

Agent/Cluster

Specifies the Agent or Cluster that controls the current Task OS.

Task OS Properties

Contains properties specific for a Task OS:

- Command Line - the command line executed when the current Task OS is executed.

Task OS Run As

Contains properties for running a Task OS as a different user:

- Username - the username with which the task is executed
- Password - the password with which the task is executed
- Domain - the domain where the task is executed

8.3.3. Task Trigger Definition

Task Trigger Definition Tab description.

Task Trigger Properties

Contains properties specific for a Task Trigger :

- Expected status - the status with which the task execution will finish.
- Time to sleep - the time used by the task to sleep before returning from execution with the desired status.

8.3.4. Task Notification Email Definition

Task Notification Email Definition Tab description.

Task Notification Email Properties

Contains properties specific for a Task Notification Email :

- From - will appear in the E-Mail From field: the entity that sent the E-Mail.
- To - the destination of the E-Mail.
- Cc - the carbon copy destination of the E-Mail.
- Bcc - the blind carbon copy destination of the E-Mail.
- Subject - the subject of the E-Mail that will be send.
- Body - the body of the E-Mail that will be send.
- Attachment path - the path to the attachment that will be included in the email

Task Notification Email Configuration

Specifies the Configuration for a Task Notification Email:

8.3.5. Task Notification File Definition

Task Notification File Definition Tab description.

Task Notification File Properties

Contains properties specific for a Task Notification File :

- Filename - the full path of the file.

8.3.6. Task File Manipulation Definition

Task File Manipulation Definition Tab description.

Task File Manipulation Properties

Contains properties specific for a Task File Manipulation:

- Operations
  - Copy
    - Source: file or directory that will be copied.
    - Destination: file or directory where the source will be copied.
    - Options
      - Overwrite: overwrite the file or the directory.
  - Move
    - Source: file or directory that will be moved.
    - Destination: file or directory where the source will be moved.
    - Options
      - Overwrite: overwrite the file or the directory.
  - Delete
    - Source: file or directory that will be deleted.
    - Options
      - Recursive: recursively delete the files in the directory

8.3.7. Task FTP File Definition

Task FTP File Definition Tab description.

Task FTP File Properties

Contains properties specific for a Task FTP File :

- Source - the source that will be copied.
- Destination - the destination where the source will be copied.
- Configuration - contans a list of available FTP Configurations.

8.3.8. User-Defined Task

Description

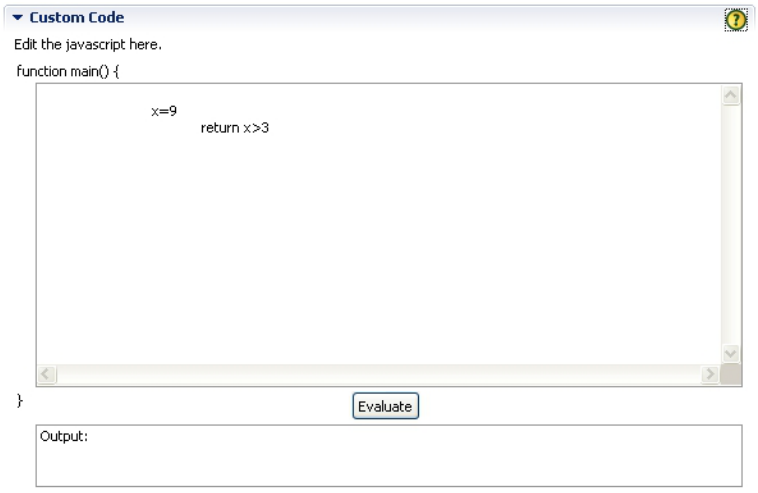
A User-Defined Task contains Javascript code which is evaluated by an agent's Service Execution. Moreover, a user fills the 'Main' function and returns a value which will either evaluate to Success or Failure.

Any exceptions encountered will result in a Failure state.

Properties

- Javascript code: the javascript code that will be executed by the event. Must return a boolean value. This value will be translated into an execution status (Success/Failure)

Examples



8.3.9. SQL Task

Description

The SQL task connects to a database using a SQL configuration and executes a statement.

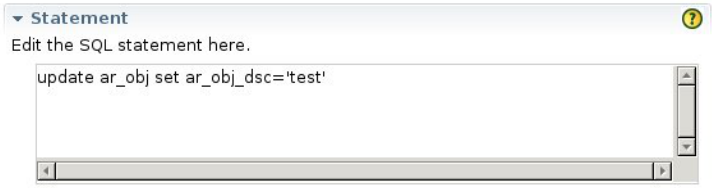
A statement can be an update, an insert, a DDL statement or a stored procedure call.

Properties

- Configuration: SQL configuration
- SQL statement: the statement to execute

Examples

The task statement section:



8.3.10. SSIS Task

Description

The SSIS task is a connector for Microsoft SSIS that comes with MSSQL 2005. It can execute packages from a SQL Server, File system or a SSIS Package Store source.

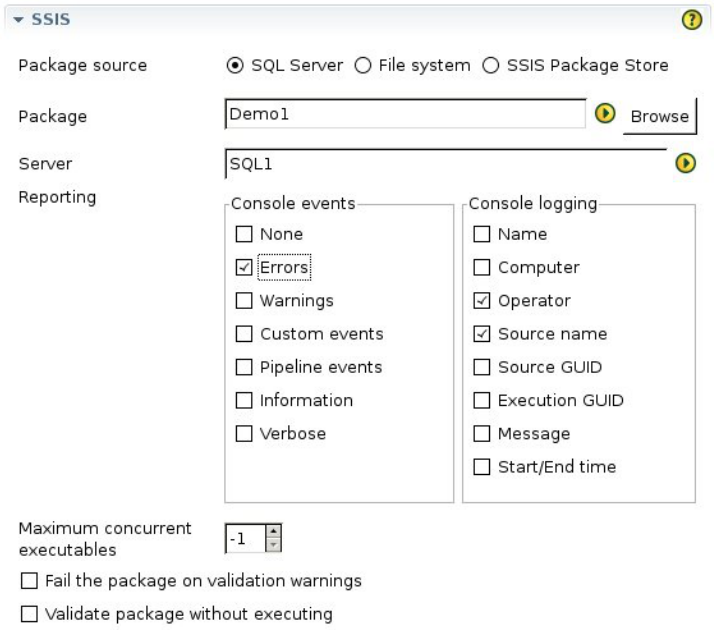
In order to integrate with Microsoft SSIS, the agent must run on the same server where the Microsoft SSIS software is installed.

Properties

- Package source: the type of package source (SQL Server, File System or SSIS Package Store)
- Package: name of the package to be executed
- Server: the SQL Server or the SSIS Package Store name where package is located
- Reporting: the level of event and logging
- Maximum concurrent executables: maximum number of running instances of a package
- Fail the package on validation warnings
- Validate package without executing

Examples

The ssis section:



8.3.11. Task Business Objects 3.1 Data Services Definition

Task Business Objects 3.1 Data Services Definition Tab description.

Task Business Objects 3.1 Data Services Properties

Contains properties specific for a Task Business Objects 3.1 Data Services:

- Configuration - contans a list of available Business Objects 3.1 Data Services Configurations.
- Job Name - the name of the job that will be executed on the Business Objects 3.1 Data Services server.

Task Business Objects 3.1 Data Services Global Variables

Contains a list of Global Variables defined for a Task Business Objects 3.1 Data Services.

A Global Variable name should be prefixed with "$" as defined in Business Objects 3.1 DS (ex: $stringVar).

A String Global Variable value should be quoted (ex: $stringVar='abc').

All other non-String Global Variables should not be quoted (ex: $intVar=123).

8.3.12. Task Business Objects 6.5 Definition

Task Business Objects 6.5 Definition Tab description.

Agent/Cluster

Specifies the Agent or Cluster that controls the current Task Business Objects 6.5.

Task Business Objects Properties

Contains properties specific for a Task Business Objects:

- Document Name- the Name of the Business Objects Document
- Domain Name - the Domain Name for the current Task Business Objects
- Configuration - the corresponding Business Objects Configuration
- Refresh

Publish

Specifies the Group where is published the current Business Objects Document, if the Publish is set to **On**.

Save
The current Business Objects document can be saved in one of the following formats:

- PDF - if the **Save** for this format in **On**, this field specifies the full path and the file name for the saved document in PDF format
- XLS - if the **Save** for this format in **On**, this field specifies the full path and the file name for the saved document in XLS format

8.3.13. Task Business Objects DI Definition

Task Business Objects DI Definition Tab description.

Task Business Objects DI Properties

Contains properties specific for a Task Business Objects DI :

- Configuration - contans a list of available Business Objects DI Configurations.
- Job Name - the name of the job that will be executed on the Business Objects DI server.

Task Business Objects DI Global Variables

Contains a list of Global Variables defined for a Task Business Objects DI.

A Global Variable name should be prefixed with "$" as defined in Business Objects DI (ex: $stringVar).

A String Global Variable value should be quoted (ex: $stringVar='abc').

All other non-String Global Variables should not be quoted (ex: $intVar=123).

8.3.14. Task Business Objects XI 3.1 Definition

Task Business Objects XI 3.1 Definition Tab description.

Task Business Objects XI 3.1 Properties

Contains properties specific for a Task Business Objects XI 3.1 :

- Configuration - contans a list of available Business Objects DI Configurations.
- Path - the path of the document that will be scheduled on the Business Objects XI 3.1 server.

Report Instance Format

The list of available formats for scheduling, based on the real format of the document
For each format of the document (ex Desktop Intelligence) there is an available list of scheduling instance format (in this case Desktop Intelligence, Excel and PDF).
The scheduled document instance will have this format
The list is dinamically constructed for the given document path

Export Instance Format

The list of available export formats for the chosen scheduling format.
For each format of the document and scheduling format there is an available list of exporting formats.
For a Desktop Intelligence document, scheduled in its original format, both Excel and PDF are allowed.
But if the same document is scheduled in Excel format, a PDF export will not be allowed.
The list is dinamically constructed for the given document path

Prompts

Contains a list of prompts available on the server for the document defined by the chosen path.
The list is dinamically constructed for the given document path. When the path is changed the page will refresh, making available the prompts defined on the server for this path.
The colored bullets give information on the status of the prompt in the AWF environment:

- Green - the prompt is configured and saved in AWF database.
- Yellow - the prompt is not configured nor saved in AWF database, but available on the server. When the task is saved, the value will be saved in database.
- Red - the AWF saved prompt is not available on the server. Should be deleted.

8.3.15. Task Business Objects XI R2

Task Business Objects XI R2 general description.

Task Business Objects XI R2 Definition

Task Business Objects XI R2 Definition Tab description.

Task Business Objects XI R2 Properties

Contains properties specific for a Task Business Objects XI R2

- Path - the path of the Business Objects XI R2 document.
- Configuration - contans a list of available Business Objects XI R2 Configurations.

Task Business Objects XI R2 Schedule

Options to schedule a Task Business Objects XI R2 in different formats :

- No schedule - do not schedule the document.
- Original Output;- the option to schedule and save the result of schedule in original format.
- PDF Output;- the option to schedule and save the result of schedule in PDF format. You have to sapcify also the path of the saved PDF document.
- XLS Output;- the option to schedule and save the result of schedule in Excel format. You have to sapcify also the path of the saved Excel document.

Task Business Objects XI R2 Prompts

Allows the user to specify the values for the prompts defined in BOXI XI R2 for the specified document

9. List Generators

Description

List generators are used to build lists at runtime and are used by looping groups.

The usage of list generators is covered here.

9.1. Counter List Generator

Description

A counter list generator takes as input a limit and will build a list of integers from 1 to the limit number (included).

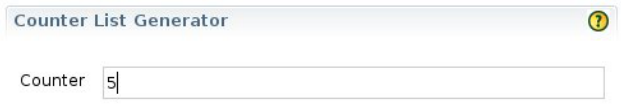
The generated lists will only have a single column.

Properties

Counter: a number greater than 0.

Examples

A counter of 5



The runtime generated list: ["1", "2", "3", "4", "5"].

9.2. Static List Generator

Description

A static list generator allows the user to insert data in a spreadsheet-like editor.

Properties

A table that contains the values assigned by the user.

Please note that the current table is limited to 20 columns and 50 rows. The character ';' is reserved therefore cannot be used inside any fields.

Examples

A table with two columns and three rows

The runtime generated list:

["Canada"; "Ottawa"]

["United States"; "Washington"]

["France"; "Paris"]

9.3. SQL List Generator

Description

A SQL list generator fetches data from a database by evaluating a user defined SQL statement.

Properties

- Configuration: SQL configuration
- SQL statement: the statement to execute which will return data
- # of columns: the number of expected columns the evaluated SQL statement will return

Warnings

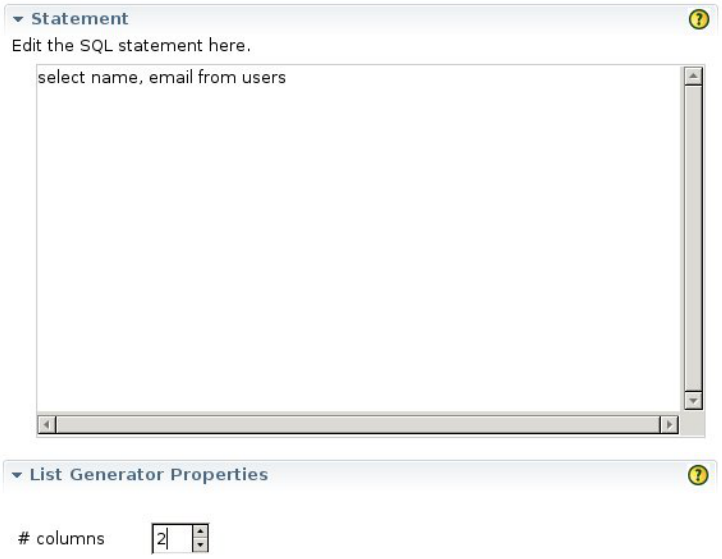
It is always preferable that the number of expected columns be equal to the number of actual returned columns.

When using a list generator in a looping context, the number of static entries mapped to a group will be equal to the expected column number. Since the data is generated at runtime, mismatches between the declared (expected) columns number and the actual number can happen.

If the actual number is smaller then the mapped entries will be blank. If it is bigger then only the first ones will be used.

Examples

A SQL statement example



The runtime generated list:

["Adrian"; "adrian@abc.com"]

["Alex"; "alex@abc.com"]

["Francois"; "francois@abc.com"]

10. Configurations

Description

A configuration is an object created to store all the required parameters to use a specific external application / utility (FTP server connection, Business Objects Credentials, etc). This configuration, allows for re-usability of these parameters, allowing for a simpler maintenance. Updating the configuration will update every task that uses it.

10.1. Transfer FTP Configuration

Transfer FTP Configuration Tab description.

Transfer FTP Configuration

Contains properties for a Transfer FTP Configuration:

- Username - the Username for the FTP connection.
- Password - the Password for the FTP connection.
- Host - host name for the FTP connection.
- Port - the port where the host is listening for FTP connection (default is 21).
- Remote Directory - full path of the directory on FTP server where the user have write access (default is root "/").

10.2. SQL Configuration

Description

The SQL configuration contains properties about external databases.

Properties

- URL: the jdbc url; database dependent
- Driver: the database type
- Username: the username to connect with
- Password: the password to connect with

Examples

The configuration section:

10.3. Configuration Business Objects 3.1 Data Services

Configuration Business Objects 3.1 Data Services Tab description.

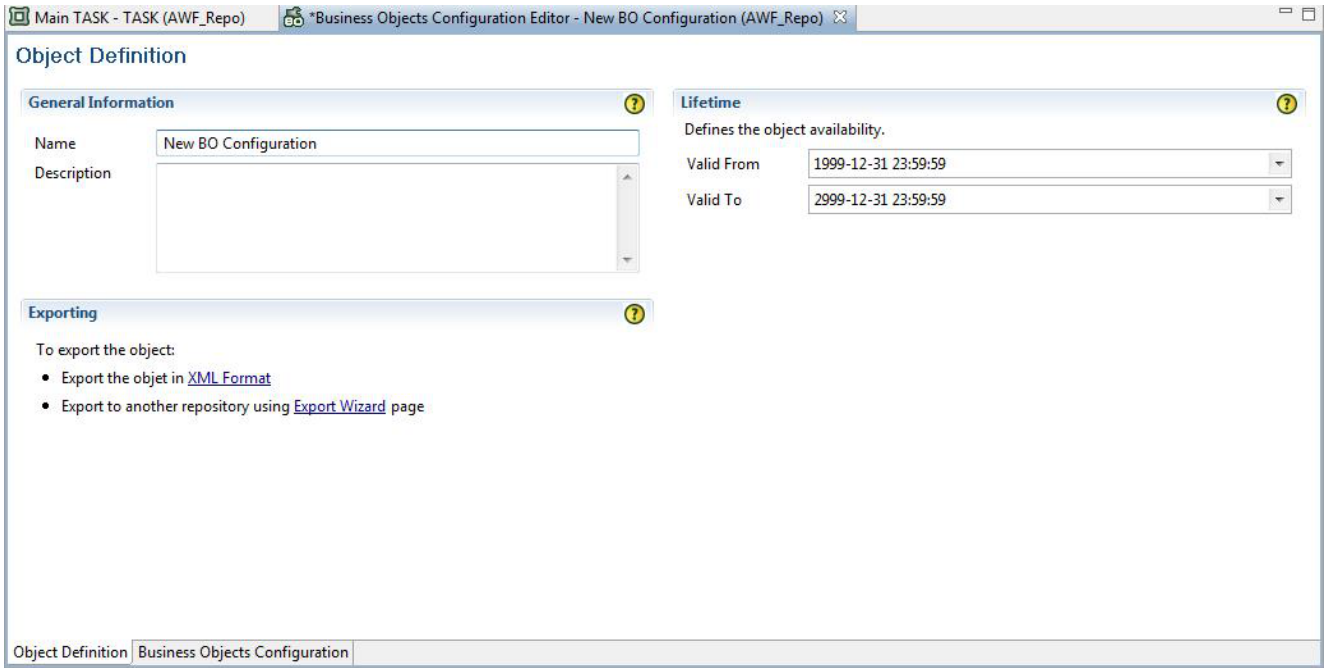
Configuration Business Objects 3.1 Data Services

Contains properties for a Configuration Business Objects 3.1 Data Services:

- Job server name - the name of the Job Server that will be used to run the job.
- Job server port - the port where the Job server listens.
- Username - the Username for the Business Objects connection.
- Password - the Password for the Business Objects connection.
- Database name - the name of the database that is used by the server.
- Database type - the type of the database used by the server.
- Repository server name - The name of the server where the repository is located, usually "localhost".

10.4. Configuration Business Objects 6.x Reference

Object Definition



General Information

- **Name :**
  The name given to this *Business Objects 6.x* configuration. This is arbitrary, but must be unique.
- **Description :**
  Free form field allowing your to store details concerning this specific configuration.

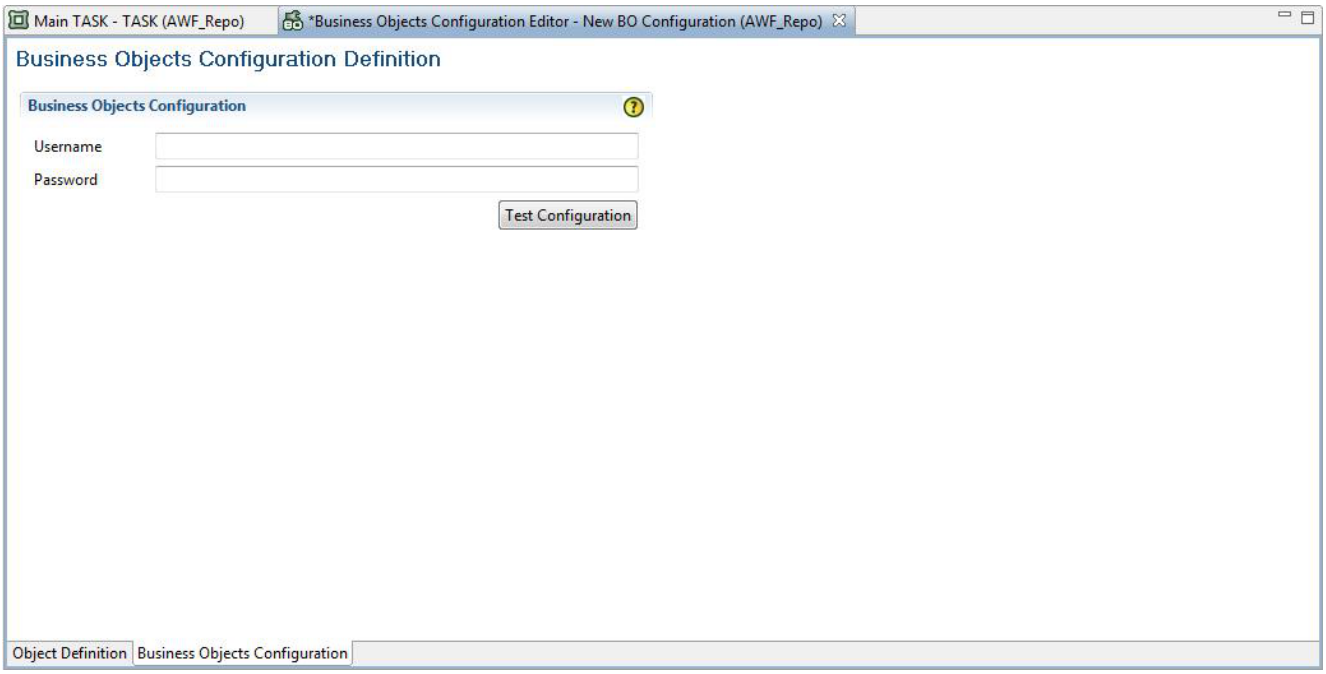
Exporting

- **XML Format :**
  Allows you to export the complete object definition in XML format.
- **Export Wizard :**
  Allows you to export the complete object definition to another repository.

Lifetime

- **Valid From / To :**
  Allows to specify the dates of validity of the current object.

Business Objects 6.x Configuration



Business Objects Configuration

- **Username :**
  Your Business Objects username.

- **Password :**
  Your Business Objects password.

10.5. Configuration Business Objects Data Integrator

Configuration Business Objects Data Integrator Tab description.

Configuration Business Objects Data Integrator

Contains properties for a Configuration Business Objects Data Integrator :

- Job server name - the name of the Job Server that will be used to run the job.
- Job server port - the port where the Job server listens.
- Username - the Username for the Business Objects Data Integrator connection.
- Password - the Password for the Business Objects Data Integrator connection.
- Database name - the name of the database that is used by Business Objects Data Integrator.
- Database type - the type of the database used by Business Objects Data Integrator.
- Repository server name - The name of the server where the repository is located, usually "localhost".

10.6. Configuration Business Objects XI 3.1

Configuration Business Objects XI 3.1 Tab description.

Configuration Business Objects XI 3.1

Contains properties for a Configuration Business Objects XI 3.1 :

- Username - the Username for the Business Objects XI 3.1 connection.
- Password - the Password for the Business Objects XI 3.1 connection.
- CMS - the name of the server CMS.

10.7. Configuration Business Objects XI R2

This section allows the administrator of a Busniess Objects XI R2 environment to specify working credentials in order to allow agile**WORKFLOW** to connect to that BO XI R2 environment and schedule tasks / verify the resulting task status.

Configuration Business Objects XI R2

Contains properties for a Configuration Business Objects XI R2

- Username - the Username for the Business Objects XI R2 connection.
- Password - the Password for the Business Objects XI R2 connection.
- CMS - the CMS used by Business Objects XI R2 connection to connect to.

The 'Test Connection' button allows you to validate the entered configuration; it's not required to save the configuration before testing it.