

# CronKit v1.1

for  Appexchange

## Installation Guide

Linvio, Inc. 2008

Welcome and thank you for your interest in CronKit. This document will guide you through the process of installing CronKit version 1.1 and configuring it to run scheduled batch jobs inside Salesforce.

## New Features:

Linvio has added the following to CronKit in version 1.1.

- Ready-to-Go Batch Jobs – We’ve added fields to the cron\_\_Batch\_Run\_\_c object and included a batch run trigger to enable scheduled clean-up of Task, Event, and cron\_\_Batch\_Run\_\_c objects using CronKit. No Apex coding is required to run these – just click-to-configure.
- CronKit Console – This is a new tab and special s-control that displays information on current batch jobs (next scheduled run, last run, status) and provides a link to the CronKit Config object for access to settings.
- CronKit Config Object – A new object used to store CronKit settings. Currently the only setting is ‘Enable CronKit Batch Trigger’, a flag that enables or disables the trigger that executes the built-in batch jobs. Your Salesforce developers can make use this object by adding fields for configuring your own custom batch jobs as well.

## Installation

### Installing Version 1.1

- Go the CronKit listing on the AppExchange and click “Get It Now”
- Provide your contact information and click on Submit
- Enter your Salesforce username, password, and select whether you are a Salesforce Administrator, Salesforce User or a Free 30 Day Trial User. When finished, click “Continue”
- Destination where you would like to install the application – in your production account (this includes Developer Edition accounts) or in your sandbox. Once you’ve read though the terms of service, click “Continue”.

### Review Package Components, Set Security Levels & Install

- Before installing you will have the option to review all the components of Cosmic Portal that will be added to your Salesforce account. The package name, version and description will also be noted. Simply click “Continue” to proceed.
- Cosmic Portal requires access to your salesforce.com objects through the API. Please review and approve the required access by clicking “Next”.
- Choose security levels to determine which user profiles can access Cosmic Portal, and click “Next”.
- Now you’re ready to install the applications. Click “Install” and CronKit will be added to your salesforce.com account. Note: You should be able to install CronKit without having to

check the box labeled “*Ignore Apex test failures that may cause the installed application not to function properly.*”

## Completing the Installation and Exposing Custom Objects

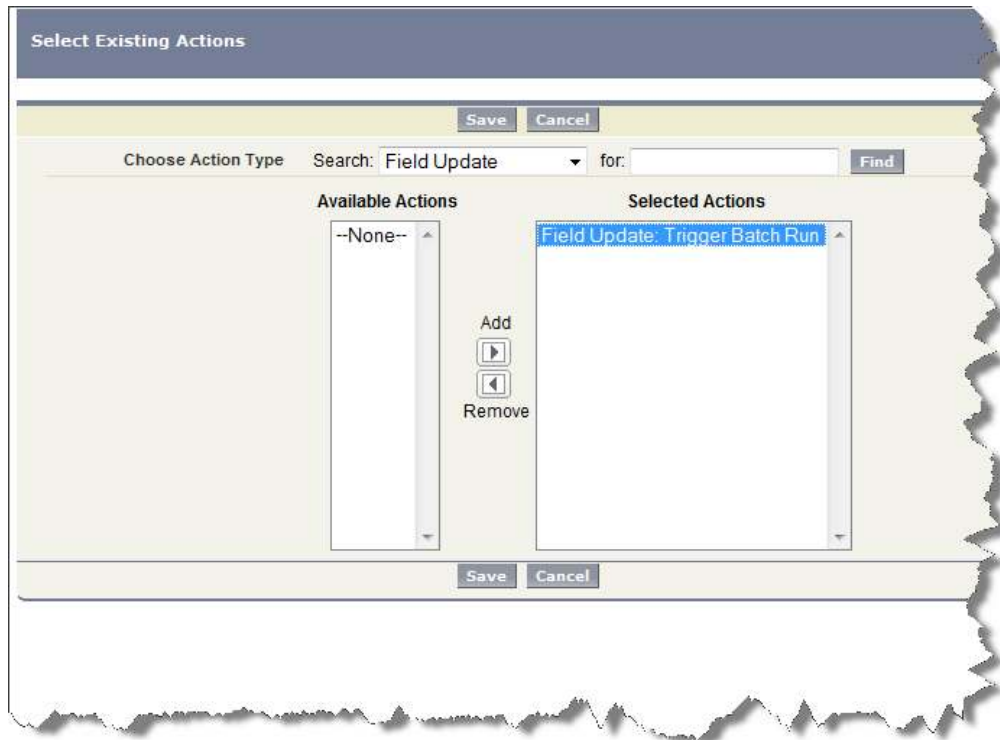
- Now that the application has been installed you have the option to immediately deploy the Custom Object components or choose to customize them first before rolling them out.
- To proceed, click “Deploy”.
- You’ve now completed the initial installation of CronKit and you will see a summary screen that displays the application details. In addition “CronKit” now appears in the AppExchange drop-down menu in the upper right-hand corner of your salesforce.com account.

## Setting up the Scheduling Workflow

The CronKit application includes two sample workflow rules that must be cloned and configured before batch job scheduling can be enabled. The sample scheduler workflow rules should be “inactive” – if they are not, please deactivate them before proceeding.

- Go to Setup > Create > Workflow & Approvals > Workflow Rules
- Click on Sample Scheduler 1 and click on the “Clone” button.
- Rename the rule to “CronKit Scheduler 1”
- Add a description (optional): “Works with CronKit Scheduler 2 to queue batch runs in time-based workflow.”
- Click “Save & Next”
- Click “Add Time Trigger” and create a workflow rule that fires **0 Hours After Batch Run: Scheduled To Run**

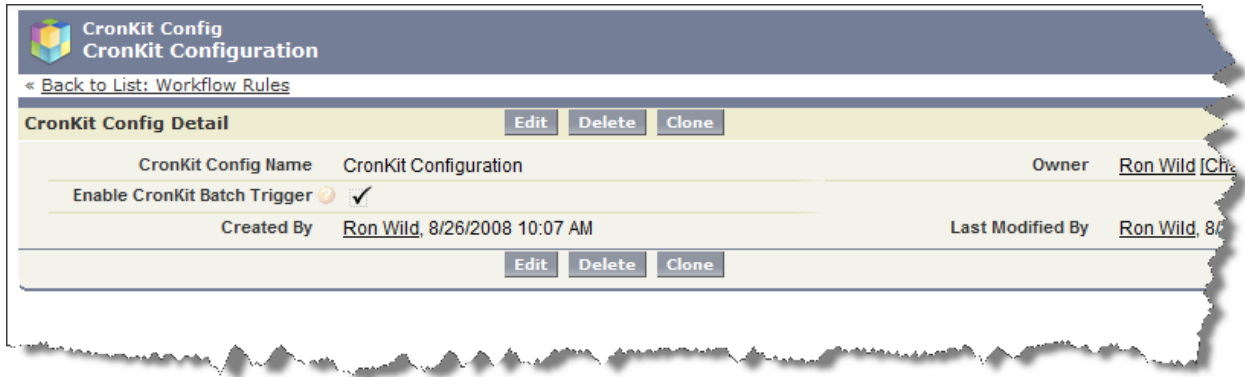
- Next we’ll add a workflow action.
- Click “Add Workflow Action” and choose “Select Existing Action”
- Choose the Field Update action **Trigger Batch Run** and save your selection.



- Click “Done” to save the new workflow rule.
- Your workflow rule page should look something like this:







As of CronKit version 1.1 there is only one setting field – Enable CronKit Batch Trigger. This box must be checked if you want to run the ready-to-run batch jobs that come with CronKit (Task, Event and Batch Run cleanup jobs).

## Setting Up a Batch Job

Configuring a ready-to-run batch job is relatively straightforward. Since the batch run records from all your other batch jobs may start to pile up, we'll create a cleanup job that will remove batch runs over a month old automatically each week.

- Select the Batch Jobs tab and click “New”
- Provide a name for the batch job
- Select “CronKit Cleanup Script” in the Batch Job Type dropdown.
- Pick a run interval (e.g. every 7 Days)
- Set **Records to Clean** to “Completed CronKit Batch Runs”
- Set the **Date Filter Field** to “Completed\_\_c”
- Set the **Age Cutoff Value** to 30 days.
- Since we want to remove all batch runs older than 30 days, set **Name/Subject Filter** to “— Any Name Value—”
- And since we are only cleaning Batch Run records, we'll leave the **Task Status Filter** field empty.
- Click “Save”

**Batch Job**  
Month-Old Batch Run Cleanup

Printable View | Customize Page | Help for this Page

Back to List: Workflow Rules

Batch Runs [2] | Open Activities [0] | Activity History [0]

**Batch Job Detail** [Edit] [Delete] [Clone] [Start Batch Run]

Batch Job Name: Month-Old Batch Run Cleanup  
Run Every: 1  
Description: Removes batch runs over a month old.

Batch Job Type: CronKit Cleanup Script  
Day/Week: Hours  
Owner: Ron Wild [Change]

**CronKit Cleanup Script Parameters**

Records to Clean: Completed CronKit Batch Runs  
Date Filter Field: Completed\_\_c  
Age Cutoff: 30

Name/Subject Filter: --Any Name Value--  
Name String or Substring  
Task Status Filter

**Stats**

Completed Runs: 1  
Unfinished Runs: 1  
Created By: Ron Wild, 8/26/2008 10:54 AM  
Last Modified By: Ron Wild, 8/26/2008 10:57 AM

[Edit] [Delete] [Clone] [Start Batch Run]

**Batch Runs** [New Batch Run] [Delete Selected] [Batch Runs Help]

Action	Batch Run Id	Scheduled To Run	Completed	Run Every	Hour/Day	Schedule Status	Status	Result
[Edit] [Del]	00000	8/26/2008 11:57 AM		1	Hours		Scheduled	
[Edit] [Del]	00001	8/26/2008 10:57 AM	8/26/2008 10:57 AM	1	Hours	--	Completed	Success

To run the job, just click on “Start Batch Run” at the top of the page. If your scheduler workflow rules are set up, and you have checked “Enable CronKit Batch Trigger”, the cleanup script will run, and queue up to run in another 7 days. Results from the batch run can be viewed by opening the last completed item in the batch run related records list.

## Upgrading CronKit

### For Current CronKit Users:

CronKit is a managed Salesforce application, so upgrading should be a relatively easy process for existing users – no uninstall/reinstall is required to move to version 1.1. However, if you are running your own custom batch run trigger(s), you may need to make a minor modification to your Apex code before enabling the Ready-to-Go batch jobs in this release:

Your batch trigger template (provided with version 1.0 should be updated to use the new `cron__Batch_Job_Type__c` object to determine whether your script is to run or not.

**See lines 12-14, 48 & 67 in the sample batch run trigger at the end of this document.**

As a suggested practice you should add your own Batch Job Type(s) to the `cron__Batch_Job_Type__c` picklist, and use the picklist value to determine which trigger will run – as there may be more than one trigger on the Batch Run object. *Version 1.1 of CronKit has its own instance of the batch run trigger that handles batch jobs with `Batch_Job_Type__c` = “CronKit Cleanup Script”.*

NOTE: The new version of the sample batch run trigger also has a couple of fixes to that improve the accuracy of scheduling (accounting for possible Salesforce workflow delays) and reporting of completion times (See lines 29 & 55 in the sample batch run trigger).

## Adding New Tabs and Fields

If you are upgrading from a previous version of CronKit new custom tabs and fields will have to be added manually. These are not automatically added by Salesforce.



To add the new CronKit tab to your CronKit application go to Setup > Create > Apps and select the CronKit application link. Then simply select CronKit from the list of available tabs and move it to the Selected Tabs list.



To add the new fields to your Batch Job and Batch run objects, go to Setup > Create > Objects and click on each the object names. Inside each object definition, find the page layouts area and edit the layouts to match the following screens:

### Batch Job Page Layout

**Legend**

- Not in Page Layout
- Used in Page Layout
- Selected
- Required
- Read-Only
- Dependent
- Controller
- Custom S-Control
- Page
- Always Displayed

**View:** Batch Job Fields

**Batch Job Fields [ Page 1/1 ]**

Age Cutoff	Batch Job Name
Batch Job Type	Completed Runs
Created By	Date Filter F...
Day/Week	Description
Last Modified By	Name/Subject...
Name String o...	Owner
Records to Clean	Run Every
Task Status F...	Unfinished Runs

### Batch Run Page Layout

**Legend**

- Not in Page Layout
- Used in Page Layout
- Selected
- Required
- Read-Only
- Dependent
- Controller
- Custom S-Control
- Page
- Always Displayed

**View:** Batch Run Fields

**Batch Run Fields [ Page 1/1 ]**

Batch Job	Batch Job Name
-----------	----------------



## Apex Code – Sample Batch Run Trigger

```
1: // This Apex trigger is designed to fire when the batch workflow scheduler
2: // checks the Trigger Batch Run checkbox or when changes are made to the Batch Run
3: // record manually.
4:
5:
6: Boolean error = false; // Var used by each batch job to flag and return an error to the Batch Run
object.
7: String results = ''; // Batch job results, also returned to the Batch Run object.
8:
9: for (cron__Batch_Run__c batchRun : Trigger.new) {
10: System.debug(batchRun);
11:
12: // Skip batch jobs not handled by this trigger
13: if (batchRun.cron__Batch_Job_Type__c == null) continue;
14: if (batchRun.cron__Batch_Job_Type__c != '---- insert user-defined batch job type ----') continue;
15:
16: if ( batchRun.cron__Completed__c != null) {
17: System.debug('Job is already completed');
18: continue; // Job has already run, skip all this
19:
20: }
21:
22:
23: if ( batchRun.cron__Trigger_Batch_Run__c == true ) {
24:
25: System.debug('Trigger Batch Run set. Running batch job.');
```

```
26:
27: // ----- Batch Job Housekeeping -----
28: //Datetime lastrun = Datetime.now();
29: Datetime lastrun =
(batchRun.cron__Scheduled_To_Run__c==null?Datetime.now():batchRun.cron__Scheduled_To_Run__c);
30: Datetime nextrun;
31: System.debug('Last run '+lastrun);
32: if(batchRun.cron__Run_Every_Units__c == 'Days') {
33: nextrun = lastrun.addDays(batchRun.cron__Run_Every__c.intValue());
34: } else {
```

```

35:             nextrun = lastrun.addHours(batchRun.cron__Run_Every__c.intValue());
36:         }
37:         if (nextrun < Datetime.now()) {
38:             nextrun = Datetime.now();
39:         }
40:
41:         // Create the next Batch Run and configure it so that the scheduler workflow
42:         // adds a Trigger_Batch_Run field update in the time-based workflow queue.
43:         cron__Batch_Run__c newRun = new cron__Batch_Run__c(
44:             cron__Scheduled_To_Run__c = nextrun,
45:             cron__Trigger_Batch_Run__c = false,
46:             cron__Batch_Job_Name__c = batchRun.cron__Batch_Job_Name__c,
47:             cron__Batch_Job__c = batchRun.cron__Batch_Job__c,
48:             cron__Batch_Job_Type__c = batchRun.cron__Batch_Job_Type__c,
49:             cron__Run_Every__c = batchRun.cron__Run_Every__c,
50:             cron__Run_Every_Units__c = batchRun.cron__Run_Every_Units__c,
51:             cron__Trigger_Scheduler_1__c = true);
52:         insert newRun;
53:
54:         // Update the current Batch Run dates and uncheck batch job trigger
55:         batchRun.cron__Completed__c = Datetime.now();
56:         if (batchRun.cron__Scheduled_To_Run__c == null) {
57:             batchRun.cron__Scheduled_To_Run__c = lastrun;
58:         }
59:         batchRun.cron__Trigger_Batch_Run__c = false;
60:
61:         // ----- End Batch Job Housekeeping -----
62:
63:
64:         // ----- Begin batch jobs -----
65:
66:
67:         if (batchRun.cron__Batch_Job_Type__c == '---- insert user-defined batch job type ----') {
68:             error = false;
69:             results = '';
70:
71:             // Insert your Apex code here... be sure to set vars 'error' and 'results' to
72:             // pass batch results back to the Batch Run object.
73:
74:         }
75:

```

```
76:         // ----- End batch jobs -----
77:
78:
79:         // Report Governor Limit Stats and set return values
80:         String limitText = 'Aggregate Queries: '+
81:             Limits.getAggregateQueries() + '/' +
82:             Limits.getLimitAggregateQueries();
83:         limitText += '\nSOQL Queries: '+
84:             Limits.getQueries() + '/' +
85:             Limits.getLimitQueries();
86:         limitText += '\nQuery Rows: '+
87:             Limits.getQueryRows() + '/' +
88:             Limits.getLimitQueryRows();
89:         limitText += '\nDML Statements: '+
90:             Limits.getDMLStatements() + '/' +
91:             Limits.getLimitDMLStatements();
92:         System.debug(limitText);
93:
94:         batchRun.cron__Results__c = results;
95:         batchRun.cron__Results__c += '\n\n'+limitText;
96:         if (error) {
97:             // write error to batch run notes field and set error flag
98:             batchRun.cron__Result__c = 'Error';
99:         } else {
100:             batchRun.cron__Result__c = 'Success';
101:         }
102:
103:     } else { // end if trigger batch job flag set
104:         System.debug('Refreshing time-based workflow queue');
105:         // Alternate Trigger Scheduler flags to keep workflow queued and current
106:         if (batchRun.cron__Trigger_Scheduler_1__c == false) {
107:             batchRun.cron__Trigger_Scheduler_1__c = true;
108:             batchRun.cron__Trigger_Scheduler_2__c = false;
109:         } else {
110:             batchRun.cron__Trigger_Scheduler_1__c = false;
111:             batchRun.cron__Trigger_Scheduler_2__c = true;
112:         }
113:
114:     }
115:
116: }
```

Download the latest version of this sample code at text from <http://www.linvio.com/cronkit/sampleApexCode.txt>