

# Home

17th January 2020 at 11:17am

## Sistan

Sistan is a tutorial for creating plugins in Tiddlywiki. It was written as a small chapter to be used in book developed by [Luis Gonzalez](#).

To get the online latest version visit: <http://sistan.tiddlyspot.com>

## Purpose

This tutorial shows how one can create a simple Tiddlywiki plugin. For demo purpose a task manager has been developed.

## Task manager plugin

For pedagogical purpose. The task manager plugin has been developed in three levels:

- Level 1: primary, very simple with minimum features and uses **Tinka** for making plugin in browser
- Level 2: intermediate, has more features and uses intermediate scripting skills and developed using **Tinka**
- Level 3: advanced, is professional one uses **Node.js** for creating the plugin

## How to study

From the sidebar, Contents tab, click on the dedicated section and follow instruction tiddler by tiddler (page by page).

1. [Concepts](#)
2. [Primary level](#)
3. [Intermediate level](#)
4. [Advanced level](#)

## Level 1: Primary

17th January 2020 at 7:59am

Contents

## Part One

## Create a task manager plugin

### Chapter contents

1. [Button to Create New Task](#)
2. [Create a List of Unfinished Tasks](#)
3. [Create a List of Finished Tasks](#)
4. [Put all Sections Together](#)

5. [Bundle as a Plugin](#)
6. [Packaging Rome as a Plugin](#)
7. [Distributing Rome Plugin](#)
8. [Using Rome Plugin](#)

# Button to Create New Task

17th January 2020 at 8:11am

Level 1: Primary

## A button

It is important to create new task in an easy and quick way. To do this a `$button` widget has been used here.

The `$button` will

- create a new tiddler to store the task as its title
- add the `task` tag to show tiddler is a task

## The script

To implement the create new task button the following code has been developed:

```
\define create-new-task()  
<$action-sendmessage $message="tm-new-tiddler" title="new task" tags="task" />  
\end  
<$button actions=<<create-new-task>> >New task</$button>
```

This script has two parts: **i.** a `$button` to create the user interface and lets user click it and create the new task and **ii.** a macro to perform the actions

## Description

- The `$button` create a button with **New task** label
- The `$button` on click calls `create-new-task` macro to perform the actions.
- The macro `create-new-task` uses the `$action-sendmessage` with `tm-new-tiddler` to create a new tiddler. This new tiddler stores the task as its title. It also set tags as `task` means the new tiddler will be tagged with `task`
- The new tiddler will be shown in screen under focus and user can simply enters the task title in the new tiddler title.

The complete code of create new task is given in [create-new-task-button](#) tiddler.

next

# Create a List of Unfinished Tasks

17th January 2020 at 8:12am

Level 1: Primary

## List tasks

The first step is to create a list of tasks. To do this a `$list` widget with proper filter is used. Each task is stored in a tiddler tagged with `task`. When a task is done, it also gets the `done` tag.

```
<$list filter="[tag[task]!tag[done]]">
</$list>
```

### Description

- The `$list` widget list tiddler based on the given filter
- The `[tag[task]!tag[done]]` selects all tiddlers tagged with `task` but NOT with `done`

## Display task title and checkbox

The task list needs to have

- a checkbox to be checked when the task is done
- the title of task to be shown

So the above code is completed as below

```
<$list filter="[tag[task]!tag[done]]">
<$checkbox tag=done/>&nbsp;
<$link to=<<currentTiddler>>{!!title}></$link><br>
</$list>
```

### Description

- The `$checkbox` widget shows a square box that is ticked (checked) when activated. It is bound to tags and when checked, adds the `done` tag to current tiddler.
- The `$link` widget creates a link to task tiddler, on click it navigates to desired tiddler.

## Empty message

It is semantic to show an empty message when the task list is empty. That means all tasks have been done.

The complete code has been given below

```
<$list filter="[tag[task]!tag[done]]" emptyMessage="You are all done! Nothing in task list">
<$checkbox tag=done/>&nbsp;
<$link to=<<currentTiddler>>{!!title}></$link><br>
</$list>
```

### Description

The `emptyMessage` is a `$list` attribute and is shown when the list is empty. Here the message "You are all done! Nothing in task list" will be shown.

The complete code of displaying list of tasks is given in [list-tasks-unfinished](#) tiddler.

[previous](#) | [next](#)

# Create a List of Finished Tasks

17th January 2020 at 8:13am

Level 1: Primary

## List of tasks done

The second step is to create a list of task have been **done**. To do this a \$list widget with proper filter is used. Each task done has the `task` and `done` tags simultaneously.

The blow code shows how finished tasks are displayed

```
<$list filter="[tag[task]tag[done]]" emptyMessage="Nothing done yet!">
<$checkbox tag=done/>&nbsp;
~~<$link to=<<currentTiddler>>{!!title}></$link>~~<br>
</$list>
```

## Description

- The \$list widget lists tiddlers based on the given filter
- The `[tag[task]tag[done]]` selects all tiddlers tagged simultaneously with `task` AND `done`

The body of \$list widget has two parts:

- The \$checkbox widget shows a square box that is ticked (checked). It is bound to tags and when unchecked, removes the `done` tag from current tiddler.
- The \$link widget creates a link to task tiddler, on click it navigates to desired tiddler. The \$link has been wrapped in `~~...~~` to format the output as strikethrough text, which implies the task has been done.

## Empty message

It is semantic to show an empty message when the finished tasks list is empty. That means no task has been finished yet.

The `emptyMessage` is a \$list attribute and is shown when the list is empty. Here the message "Nothing done yet!" will be shown.

The complete code of displaying list of tasks is given in [list-tasks-finished](#) tiddler.

[previous](#) | [next](#)

# Put all Sections Together

17th January 2020 at 8:22am

Level 1: Primary

## A complete script

This section describes how to put all elements of task manager together to have a working application.

The task manager has three elements

1. a button to create a new task. This has been developed in [create-new-task-button](#)
2. a list to display unfinished tasks, which has been developed in [list-tasks-unfinished](#)
3. a list to display finished tasks, which has been developed in [list-tasks-finished](#)

## A User Interface

A user interface in brief a UI is used to display all elements of task manager in a simple order to let user easily interact with application.

Here we also use a name for our application. We called this Level 1: Task Manager, **Rome**. So, Rome is the name of our application.

To create the interface and put all elements together, a new tiddler is required as below

- title: Task Manager Level 1: Rome
- text: all sections of task manager in proper order as below
  - the button to create new tasks
  - the list of unfinished tasks
  - the list of finished tasks

So, at the end the body of tiddler is like below

```
! My daily tasks

{{create-new-task-button}}

!! Unfinished tasks
> {{list-tasks-unfinished}}

!! Finished tasks
> {{list-tasks-finished}}
```

Navigate to [Task Manager Level 1: Rome](#) to see the working application.

[previous](#) | [next](#)

## Bundle as a Plugin

17th January 2020 at 8:31am

Level 1: Primary

There are several different methods to bundle an application as a plugin. These methods are based on

1. creating and packaging in browser
2. creating and using the Node.js

Here the first method which is simpler is used. To learn how Tinka works see [Packaging with Tinka](#). A short description is also given in this wiki [Using Tinka to Create Plugins](#)

[previous](#) | [next](#)

# Packaging Rome as a Plugin

17th January 2020 at 11:05am


Level 1: Primary







## Create the plugin

In this tutorial [Tinka](#) plugin packager is used. After you make sure Tinka is installed in your Tiddlywiki and working fine, then follow the below steps to create the Rome plugin.

## Step One

Goto [\\$/ControlPanel](#), open "Tinka Plugin Management" tab and then click on "Create a new plugin". Fill in the inputs as figure below. Note that the bold items are the required ones, without them Tinka will fail to create your plugin.

 [\\$/ControlPanel](#)



InfoAppearanceSettingsSavingPluginsKeyboard ShortcutsTinka Plugin Management

InstalledCreate a new PluginArchive

### Create New Plugin

Usage: Enter the necessary metadata for your plugin and use the Filter selection below to pick the tiddlers that should be added to the plugin. After selecting the tiddlers, press 'Package Plugin'. Refer to the [Documentation](#) for further help.

Step 1: Enter Metadata

**Plugin-Type:**

<b>Plugin Path:</b>	<input type="text" value="\$/plugins/kookma"/> / <input type="text" value="rome"/>		
Author:	<input type="text" value="Mohammad Rahmani"/>	Source:	<input type="text" value="http://sistan.tiddlyspot.com"/>
Dependents:	<input type="text"/>	List:	<input type="text" value="e.g. readme usage"/>
<b>Plugin Title:</b>	<input type="text" value="Rome a simple task manager"/>		
Version:	<input type="text" value="0.1.0"/>	Core-Version:	<input type="text" value="5.1.21"/>

**bold** = required field

Step 2: Add Tiddlers

**Figure 1. Create Rome plugin using Tinka. Step 1, enter the plugin information.**

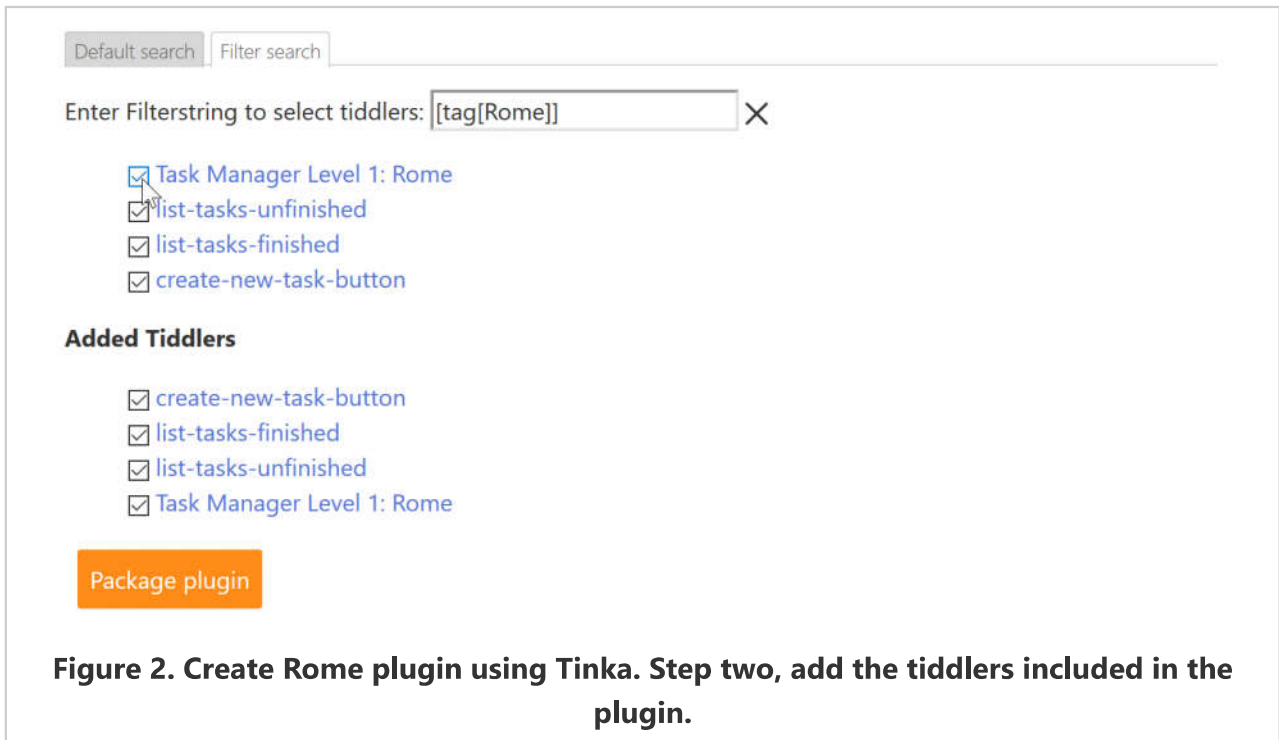
## Step Two

In this step, add all tiddlers required for creating plugin. In this Level 1: Task Manager example, all these tiddlers are tagged with Rome, so they are:

1. **create-new-task-button**
2. **list-tasks-finished**
3. **list-tasks-unfinished**
4. **Task Manager Level 1: Rome**

To add these tiddlers to your **Rome** plugin, continue with step two. Use the **Filter search** with `[tag[Rome]]` to list all tiddlers included in Rome plugin. Then click the provided checkbox to add these tiddlers to plugin.

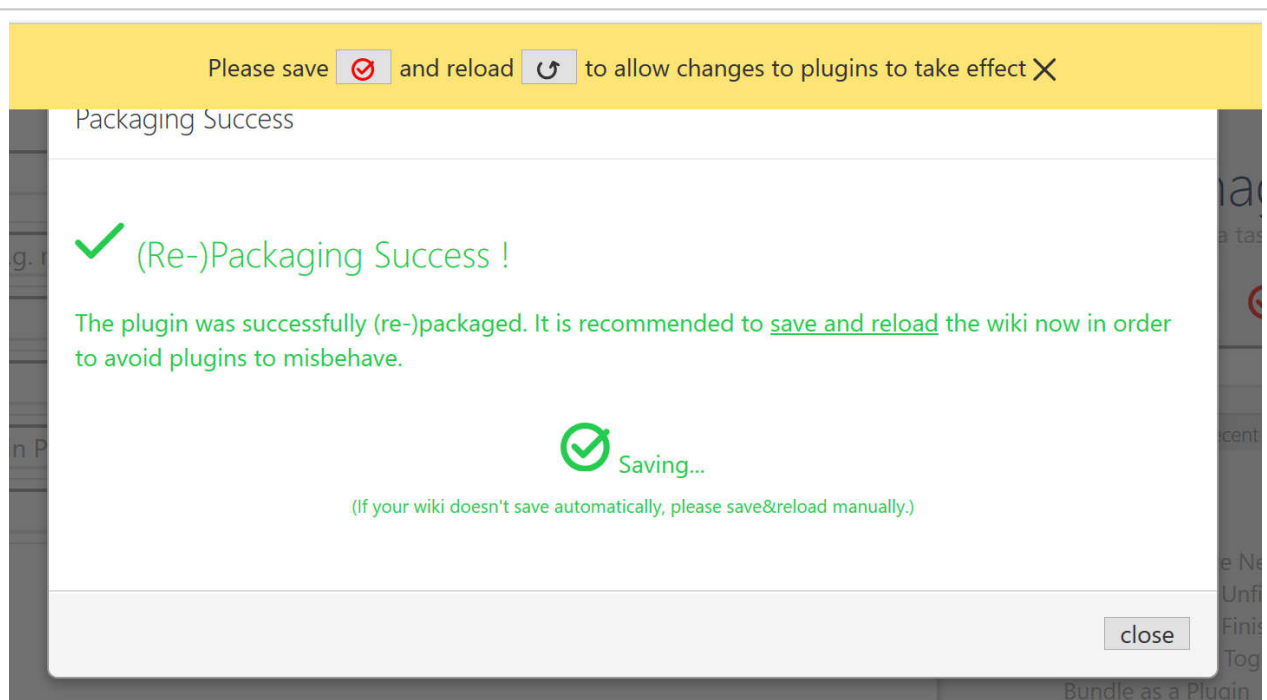
The figure below shows how step two looks like.



**Figure 2. Create Rome plugin using Tinka. Step two, add the tiddlers included in the plugin.**

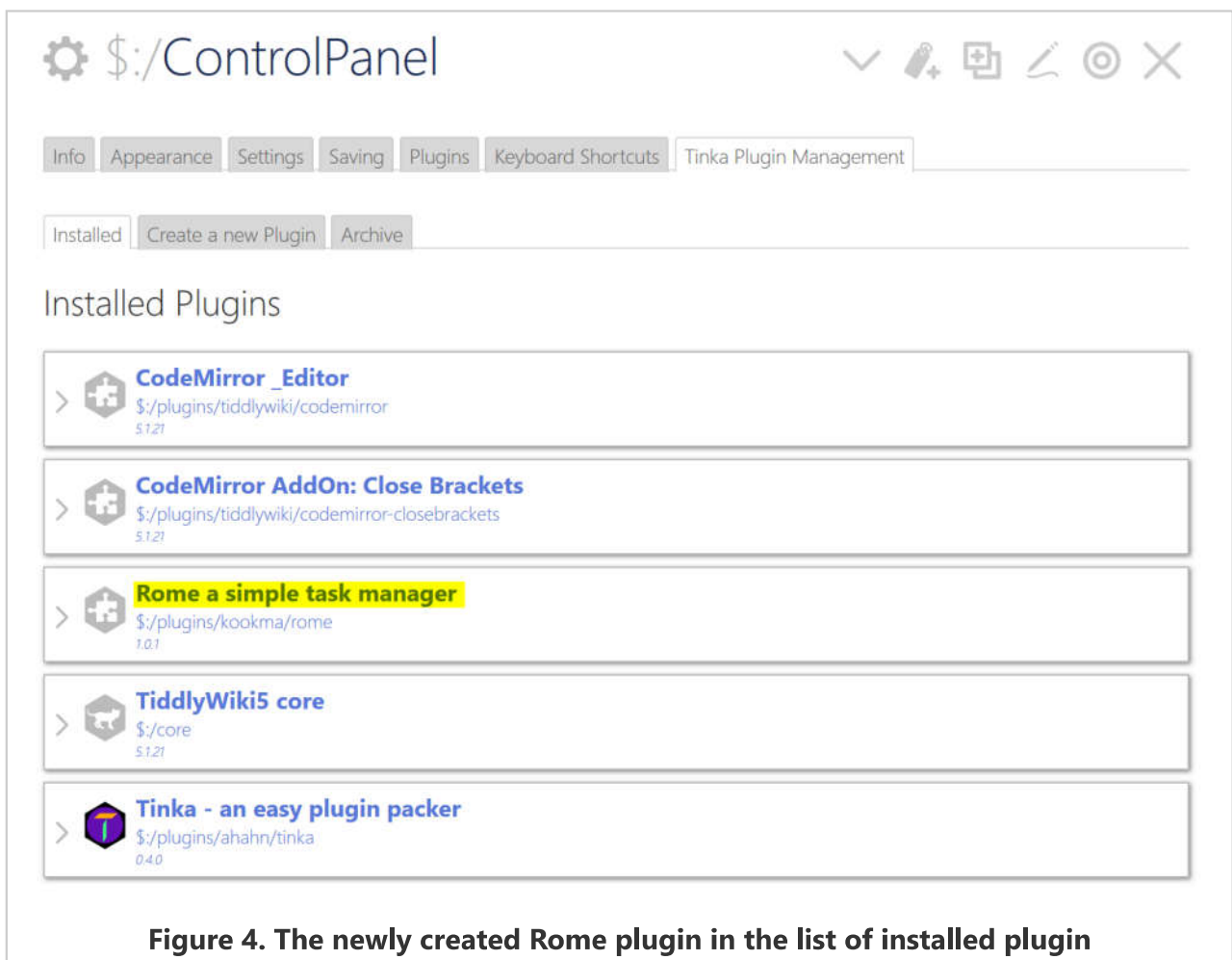
## Final step

The next step is to click the **Package plugin** shown in Figure 2 above. After that Tinka will show a message as Figure 3. indicating the plugin has created successfully. Save the wiki and reload.



**Figure 3. Tinka message after finished creating plugin.**

To make sure the plugin has been created, open the [\\$/ControlPanel](#), goto **Plugins** tab and find the new plugin in the list of installed plugin. If everything goes well you should see the new Rome plugin in the list as below.



**Figure 4. The newly created Rome plugin in the list of installed plugin**



# Distributing Rome Plugin

17th January 2020 at 11:24am

Level 1: Primary

The simplest way is to drag and drop the plugin tiddler from source wiki to target wiki. So for Rome plugin you can

## First method

- Goto [\\$/ControlPanel](#)
- Open the **Plugins** tab and drag and drop the **Rome a simple task manager** to target wiki

## Second method

- Create a link to plugin in a readme or home tiddler
  - for Rome it is [\\$/plugins/kookma/rome](#)
- Drag and drop the plugin tiddler link to the target wiki

There are more advanced methods will be discusses in next parts of creating Task Manager Plugin.

[previous](#) | [next](#)

# Using Rome Plugin

17th January 2020 at 11:29am

Level 1: Primary

After installing the Rome plugin in your Tiddlywiki, you can simply embed its interface whenever you like.

As an example here, a new tiddler called [My First Plugin: Task manager](#) was created with the below content:

```
{{Task Manager Level 1: Rome}}
```

## Use task manager in sidebar

If you like to use the Rome task manager in the sidebar do as following

1. Open [My First Plugin: Task manager](#)
2. Tagg it with `$/tags/SideBar`
3. Add the caption field with `My Tasks` as value

Now look at the sidebar and see the new tab **My Tasks**.

[previous](#)