

# Getting started with linkalist

15/01/2018



## What is linkalist?

Linkalist is a general-purpose data management system that allows you to create data driven applications for the web without having to write any code. We also provide SDKs for both iOS and Android to simplify data access from mobile devices. However, as our web applications are fully responsive, you don't need to get involved in mobile application development if you don't want to.

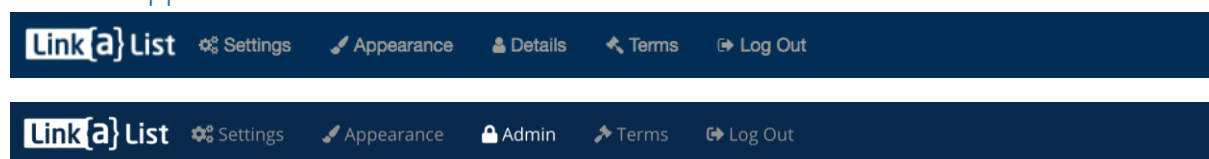
Linkalist can be used to store data in its own database, to interact with your own micro-services or to operate in a hybrid mode where you primarily use linkalist for storage but implement logic particular to your application using micro-services.

## Introduction

This document describes what you need to do to get started with using linkalist to define your web or mobile application. Before we go into more detail, we should define a few terms used in the context of this document and linkalist in general.

- **Application:** This is a collection of lists whose relationships describe the behaviour of your data. Think of an application as being something similar to a mobile phone app.
- **List:** Within linkalist, a list is defined as a collection of objects. You may define your object with as many fields as you like
- **Field:** A field is a single piece of data that can be stored as part of a list item.
- **View:** A view describes how a list appears to an application's end users.
- **Column:** A (view) column describes how a field appears to the application's end users.

## General Application Structure

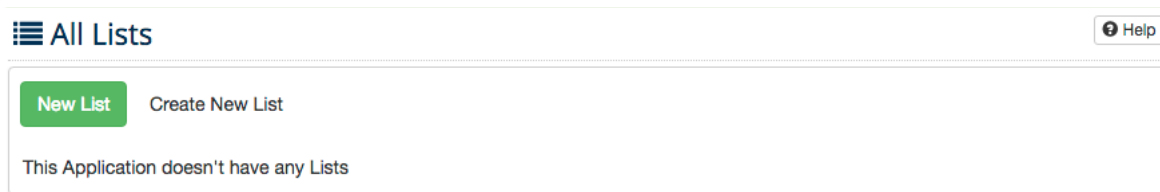


When you first log in, you'll see that the menu bar has six main areas. These are the general areas for the application and have the following purpose

- **Editor:** The area where your users will spend most of their time – this is the area in which you interact with your application. You get to the editor area by clicking on your application name or logo (in this case – the link{a}list logo).
- **Settings:** This is where you define the application list structures and how they are presented to the user as views. This area is only accessible to manager-level users and nobody with a lower level will see this menu item.
- **Appearance:** This where you define how your application appears – you can manipulate the colour palette and the CSS styles used within your application here. This is only accessible to managers.
- **Details:** This is where you edit your account details and for manager-level users, where you can edit your application details
- **Terms:** Display the T&Cs for your application
- **Log Out:** Log out of your application

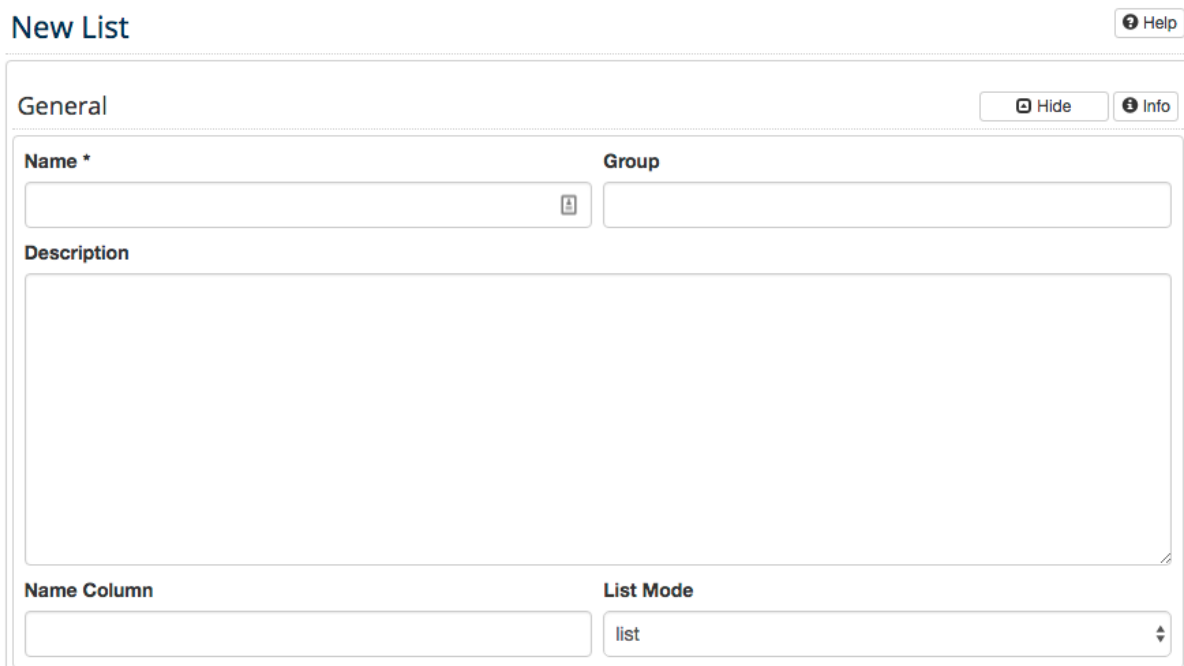
## Creating a List

The first thing you need to do is create a list for your application. This is done from the settings tab. Click [here](#) and then click the “New List” button to get started.



Your next step is to set the various details on your list. The most important thing is the name of your list. This must be unique across your application. The pattern we use in linklist is to **name this as a singular** as this keeps things consistent. Hence if your list is to contain notes, call it note and the system will create a database table called notes and references will be note\_id. This leads to a bit of grammatical strangeness where a work ends in s or y but this can't be helped.

The description should describe what your list is used for and this may be presented to your customers as help information as the system developers. The name column can normally be left blank – the system will automatically a few columns within the system – one of which will be the name column.



There are six subsections with advanced settings for lists. As this is a “getting started” document, we won't go into these now. As they have sensible defaults, you can safely ignore these for now.

For the moment, you should stick with the “list” mode for the list. We also support tree structures if your application needs nested data but we'll deal with this in another document. The “Group” field is for use when a number of lists have a common theme. In this case, you can define all of these lists as having the same “Group” name and they will then be displayed together in the settings tab.

Once you press the “Save” button the system will create the list and all is good to continue

## Defining your List

### All Lists

Help

<b>New List</b> Create New List		
Name	List Mode	Data Source
First List	list	database
<b>edit</b>	delete	export
fields	documentation	views
sources	schedule	test
translations		

Now that we've created a list, the system is now in a position where you can start manipulating data. However, you'll need to edit the list to set it up the way you want it. The first thing you'll need to do is to define what fields go into your list. You do this by clicking the "fields" button for the list.

Once you go into a list, you'll see that the system has created quite a few fields for your list already. Most of these are system fields and cannot be deleted within the system. The only exception is that you may remove the "Status" field which you should do if you don't want it.

### List Fields First List

Help

<b>New Field</b>	Add a new simple data field to this list				
<b>New Reference</b>	Create a link between this list and another list				
<b>New Document</b>	Add a new document field to this list				
<b>New Image</b>	Add a new Image field to this list				
<b>New Calculation</b>	Set up a calculation to be included in this list				
Field	Type	Comment	Listed	Hidden	Mandatory
Id	Number	PK for the table	No	No	Yes
Name	Short Text	Name for the item	Yes	No	Yes

## Creating a Field

You'll see from the above that there are several different types of fields you can create. For now, let's just work on creating a simple field and we'll return to the others later. To start this process, start by pressing the "New Field" button. This takes you to a screen where you can set up the various values associated with your field.

General Hide Info

**Field Label**

**Type**

Number

**Comment**

**Length**

**Options**

More Settings Show Info

Save Cancel

There is quite a bit here to set up but for the most part it is fairly self-explanatory.

First, you need to set up a label (name) for your field. This is how that field will appear to your customers and also defines how it goes into the database, so you'll need to make this unique for the table. You should only use numbers, letters and spaces as field labels. You may not have fields differing only by the case of the characters (eg. "Name" and "name").

Next, you need to define a type for your field.

- **Number:** A signed integer between approximately -2 billion and + 2 billion.
- **Fixed Text / Short Text:** Use these for short blocks of text – you will need to define a **length** value for these that defines the maximum number of characters you can store in the system.
- **Selection:** A field that allows you define a number of options. These should be entered in the **options** box, with one option per line.
- **Long Text:** This is used for long blocks of text. Generally, you must not enter more than about 65,000 characters into one of these.
- **Date & Time:** There are various permutations of date and time.
- **Timestamp:** This allows you to timestamp your records. This is only provided for compatibility with imported databases – it is best to avoid this and use the built-in create time & update time timestamps.
- **Currency:** Use this to store a monetary value. We don't currently convert between currencies and we only support currencies with 100 sub-units in a unit. (eg cents in a euro)

- **Decimal:** Stores larger decimal numbers
- **Yes/No:** Stores a yes / no choice
- **Location:** Stores a latitude / longitude pair

After setting up the type, you'll need to define a comment for your field. This will be used by the help system and the user interface to inform your customers about what the field is for.

Length and Options are used for text types and selections as described previously.

Of the "More settings", the only ones you need be concerned about are "Mandatory" and "Default". If a field is mandatory, the user must enter something into it, unless the default is defined in which case the system will fill in the default. You can safely ignore these for now.

## Defining Views

When you create a list, the system automatically creates a view for you. In general, you will need to adjust this as it won't contain enough data to be useful. You start this process by clicking the "views" button for your list. If you prefer you can create a new view, but let's start with modifying the one you have to make it more useful. You start this by clicking the "columns" button.

The screenshot shows the 'List Views' interface for 'First List'. At the top left, there is a 'New View' button and a 'Create New View' link. Below this is a table with the following columns: Name, On Home Page, On View Menu, Is Default, and Personalised. The table contains one row for 'All First Lists' with values: No, No, Yes, and No. Below the table is a row of buttons: view, edit, delete, copy, export, columns, conditions, sorting, links, sets, styles, and groups. A second row of buttons includes: defaults, users, roles, test, change log, and translations.

This gives you a list of the columns in your list. You'll see that the system has already placed the ID field here for you. Let's leave this for now and add in the name field. Press the "New View Column" button.

The screenshot shows the 'New View Column' configuration interface for 'All First Lists (First List)'. It has a 'General' tab and 'Hide' and 'Info' buttons. The form contains the following fields:

- Title \***: An empty text input field.
- List**: A dropdown menu with 'First List' selected.
- List Field**: A dropdown menu with 'Id' selected.
- Purpose**: A dropdown menu with 'None' selected.
- View Column Group**: A dropdown menu with 'None' selected.

At a minimum, for each column, you need to give it a

- **Title:** This is the title of the field in the view and appears on the top of a list
- **List:** This is the list from which the column comes. You may use the list itself or any lists to which it refers. (More about references later)
- **List Field:** The list field to display for the column.
- **Purpose:** This is used by the more specialised views to define special purpose columns. You can ignore this for the simple table views we'll be working with for now.

There are multiple extra settings under Visibility, Styling and Miscellaneous, but for the purposes of getting started you can ignore these for now.

## Conditions in Views

Once you add conditions to views, things start getting a bit more interesting as these allow you to show lists of items that don't contain the entire list contents.

**New View Condition** All First Lists (First List) Help

**Name \***

**List**  
First List

**Model Field**  
Id

**Compare Type**  
greater than

**Compare Value**

Parsed  
 **Must Match**

**View Parameter Set**  
None

For each condition, you need to name it, choose a list and field, a comparison type and a value to compare against. The "Must Match" field allows you to define conditions where only one of several conditions need match. By default this on. You can ignore the View Parameter Set option for now.



## Editing Views

👁️ Edit View First List 🔗 Help

**General** Hide Info

**Name \***

**Description**

**Column Count \***  **Max Items**

You may go back and edit your view, or create a new one. When you do this, as a minimum you need to set a name, a description and a column count. The column count defines the number of columns you see when you view a single item belonging to the view.

If you want a view to appear on your application's home page, you need to go to the "Placement" section and select "On Home Page". You will see other options as to special placement for your view.

The bulk of the functionality of views is buried in the various options available from this page and their complexity well exceeds the scope of a "Getting Started" document. There is further documentation available detailing the various details required for setting up various more complex views. For now you can ignore this as the basic table-type views are perfectly adequate for the purposes of putting a basic app together.

## References

An application with a single list, or indeed one with two unconnected lists is a very boring thing so the last topic for this document is to set up a second list and connect it to the first list in a way that an item in the first list may have multiple second list items.

First, you need to create another list as per the first step in this document. When you're done with that, you should end up with your application lists looking something like this.

☰ All Lists Help

Name	List Mode	Data Source
☑ First List	list	database
☑ Second List	list	database

Now we need to go into the “Fields” selection of the second list and create a reference to the first list. This will construct a relationship between the lists such that any element in “First List” may have one or many “Second List” elements associated with it. So for example, if you First List represented Projects, your Second List could be Tasks. Then after this, each Project could have many tasks.

So, from the Fields list of the second list, click the “New Reference” button.

☰ New Reference Second List Help

General Hide Info

Reference List  
First List

Comment

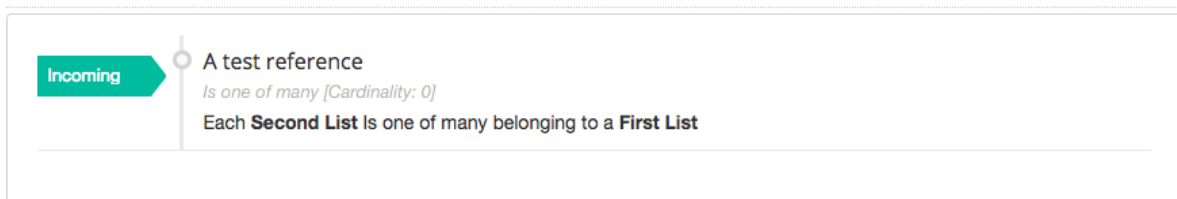
More Settings Show Info

Advanced Show Info

Save Cancel

Select the “First List” from the reference list selection, give it a comment and you're done. You need not be too bothered about More Settings or Advance, except possibly for the “Mandatory” option. After this you should end up with the following reference in “Second List”.

### List References



## Linking the Lists

### View Links All First Lists (First List)

[New View Link](#) [Create New View Link](#)

This view doesn't have any links

You now have the database structure required to link the two lists but we still have to set up the user-interface link. To do this, you need to go back to your view on the first list and set up a link.

### New View Link All First Lists (First List)

[Help](#)

<b>Name *</b>	<b>Mode</b>
<input type="text"/>	internal
<b>List</b>	
Second List	
<b>Dest View</b>	
All Second Lists	

Visibility & Appearance [Show](#) [Info](#)

External Links [Show](#) [Info](#)

Miscellaneous [Show](#) [Info](#)

[Save](#) [Cancel](#)

The first thing you need to do here is give your link a name. This describes the title of the link for both where it is referenced and when it is displayed. The mode allows for more advanced link mechanisms but for now we'll leave it at internal. The other mandatory values are the List and the Dest View which define the list to link and the view to show in the link.

By default, links are shown both on the associated list and when viewing items in the list. This and other settings can be changed in the various other sections involved in setting up a link.