
Arise Documentation

Release 2.3

Arise.io

August 21, 2013

CONTENTS

Contents:

SETUP YOUR FIRST A/B TEST

1.1 Overview

1.1.1 What is A/B testing?

Testing is a way of showing users different variations of your app, and then measuring how each variation affects your goals. For example, if you're interested in having your users make more in-app purchases, you might test the phrase "buy more coins now!" versus "save time by buying coins". Arise's A/B testing tools can tell you which one will make you more money. You can also use Arise to test new in app-features, evaluate performance and roll out changes without resubmitting your app.

1.1.2 What is a conversion?

Tests are only good when they can measure some metric and show you which path is performing better. To help out our system, that metric should be a "conversion". Good examples of goals are things like an in-app purchase, a new account created, or even a tap on an advertisement. We can then calculate the [conversion rate](#) (Number of conversion/ Number of views).

1.2 Getting Started

1.2.1 1. Create an account

If you do not have an account yet, register on the [dashboard](#). The dashboard is your tool to configure your experiments.

1.2.2 2. Create and configure your variation

Click on "Create a new experiment" to create your first experiment.

Two variations are created by default: variation A (original) and variation B (test).

Variation values

Variation values will be transmitted to your app. You can experiment:

- call to actions
- headline, product descriptions (words, style, number of words)

- in-app purchases (prices, multiple contents)
- forms (types of fields, length, layout, error handling)
- layout and design (position and grouping of content)
- images

You need to set different values for each variation. We will consider the variation A as the default variation (original). You will be able to create up to 8 variations. You can create a new variation by clicking on the blue ‘+’ button. However, you will get more reliable and faster results by testing as few variations as possible.

Distribution

You can set the distribution of your variations. The total of all the variations must always be 100%. You can for example set 70% on the variation A (original) and 30% on the variation B (test). Once your app is deployed, you will be able to view the conversion rate for each variation.

1.2.3 3. Start your experiment

Once you have configured your experiment, you will need to click on the “Start experiment” button. Once an experiment is started, experiment values are not allowed to be modified. You will also not be able to add or remove a variation. However, the distribution can still be modified. Requests from new devices will use the new distribution. Devices that have already an experiment assigned won’t change.

1.2.4 4. Install the Arise SDK in your app

We currently support iOS and Android. Please read the following documentation to install the Arise SDK in your app:

- [*Arise iOS SDK*](#)
- [*Arise Android SDK*](#)

1.2.5 5. End your experiment

Once you have significant results for an experiment, you have the choice between keeping the variation A (original) or using a test variation (B,C,etc.). All the devices will display the selected variation when you validate your choice.

ARISE IOS SDK

This document will help you to install the Arise iOS client in your application.

2.1 Installation steps

2.1.1 1. Add the Arise library to your project

First, download the [Arise SDK for iOS](#). Unzip it and drag it inside your project's Frameworks folder in XCode. Ensure that you select 'Copy items into the destination group's folder' when the dialogue box appears.

2.1.2 2. Add dependencies

In XCode:

- select your project in the project navigator
- in the project settings select your target
- click on the Build Phases tab
- Open the Link With Libraries collapsible panel
- Click on '+'
- Search for `libsqlite3.dylib` and click on the Add button.

2.1.3 3. Initialize the framework

Add the following line in your AppDelegate.h:

```
#import <Arise/Arise.h>
```

Add the following line under `application:didFinishLaunchingWithOptions` of your AppDelegate.m file to initialize the framework:

```
[Arise initializeWithKey:@"9c51b5e8f06ebd26728f29954365098f052c68c8"];
```

Replace the value of the key by your own key. You can find it at the top of your dashboard.

2.1.4 4. Get the experiment value

You can request a variation value using the `getVariation:` method with the experiment name (same as the one displayed in your dashboard). You have also to set a default value in case of no connection to the server. “Buy it now” is the default value in the following code snippet.

```
[ABTest getVariation:@"Experiment1" defaultValue:@"Buy it now" data:^(NSString *value){
    [_purchaseButton setTitle:value forState:UIControlStateNormal];
}];
```

Do not forget to import the Arise SDK in your header file:

```
#import <Arise/Arise.h>
```

The default value will only be printed if the application has never succeeded to connect to the server. We recommend to set the default value to the same value as your variation A value.

2.1.5 5. Record events

Now that you have setup your application for testing, you will need to record views and conversion events. You need to provide the experiment name as a parameter to associate the events with an experiment.

Record a view:

```
[ABTest recordView:@"Experiment1"];
```

Record a conversion:

```
[ABTest recordConversion:@"Experiment1"];
```

Views and conversions events are stored on the device until an internet connection is available. Our framework does work properly even in case of no connectivity.

2.2 Full code example

```
#import "ViewController.h"

@interface ViewController ()

@end

@implementation ViewController

- (void)viewDidLoad
{
    [super viewDidLoad];

    // Get and setup the variation
    [ABTest getVariation:@"Experiment1" defaultValue:@"Buy it now" data:^(NSString *value){
        // Use the variation value to customize our application
        // ...

        // For example :
        // Change the title of the purchase button
        [_purchaseButton setTitle:value forState:UIControlStateNormal];
    }];
}
```

```

}

- (void)onLoadPurchasePage
{
    // the user is viewing the item purchase page
    // record a view event
    [ABTest recordView:@"Experiment1"];
}

- (IBAction)onPurchase:(id)sender
{
    // the user has bought the item
    // record a conversion event
    [ABTest recordConversion:@"Experiment1"];
}

- (void)didReceiveMemoryWarning
{
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}

```

2.3 Notes

The Arise iOS SDK supports iOS 5.0 and later.

ARISE ANDROID SDK

This documentation will help you to install the Arise Android client in your application.

3.1 Installation steps

3.1.1 1. Add the Arise library to your project

First, download the [Arise library jar file](#) and drag it inside your project's /libs/ folder.

3.1.2 2. Add permissions

In your project, right click on AndroidManifest.xml Click on Open With/Android Common XML Editor. Add after the `<uses-sdk />` tag:

```
<uses-permission android:name="android.permission.INTERNET"></uses-permission>
```

3.1.3 3. Initialize the framework

In the onCreate function of your main activity, you need to initialize the framework:

```
// Initialize the arise library
String authKey = "9c51b5e8f06ebd26728f29954365098f052c68c8";
Arise.initialize(getApplicationContext(), authKey);
```

Replace the value of authKey by your own key. You can find it at the top of your dashboard.

3.1.4 4. Get the experiment value

When you plan to run the experiment, you will need to call the `getVariationWithListener` with the experiment name (same as the one displayed in your dashboard) to get the experiment data. You have also to set a default value in case of no connection to the server. "Buy it now" is the default value in the following code snippet.

```
// Get and setup the variation
ABTest.getVariationWithListener("Experiment1", "Buy it now", new VariationListener() {
    @Override
    public void onVariationAvailable(String value) {
        final String buyMessage = value;
    }
});
```

```
    }
  });
```

The default value will only be printed if the application has never succeeded to connect to the server. We recommend to set the default value to the same value as your variation A value.

3.1.5 5. Record events

Now that you have setup your application for testing, you will need to record views and conversion events. You need to provide the experiment name as a parameter to associate the events with an experiment.

Record a view:

```
ABTest.recordView("Experiment1");
```

Record a conversion:

```
ABTest.recordConversion("Experiment1");
```

Views and conversions events are stored on the device until an internet connection is available. Our framework does work properly even in case of no connectivity.

3.2 Full code example

```
package com.example.shoestore;

import io.arise.ABTest;
import io.arise.Arise;
import io.arise.VariationListener;
import android.os.Bundle;
import android.app.Activity;

public class MainActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // Initialize the arise library
        String authKey = "9c51b5e8f06ebd26728f29954365098f052c68c8";
        Arise.initialize(getApplicationContext(), authKey);

        // Get and setup the variation
        ABTest.getVariationWithListener("Experiment1", "Buy it now", new VariationListener() {
            @Override
            public void onVariationAvailable(String value) {
                // Change the button label
                final String buyMessage = value;
                // Use the buyMessage to customize our application
                // ...
            }
        });
    }
}
```

```
}

private void onLoadPurchasePage() {
    // the user is viewing the item purchase page
    // record a view event
    ABTest.recordView("Experiment1");
}

private void onPurchaseCompleted() {
    // the user has bought the item
    // record a conversion event
    ABTest.recordConversion("Experiment1");
}
}
```

3.3 Notes

The Arise Android SDK supports Android 2.3.3 (API level 10) and later.

RELEASE NOTES

4.1 Version 2.4

August 22th 2013

4.1.1 Dashboard

- Adding support up to 8 variations per experiment (A/B/N testing)
- Stunning performances (no page reload)

4.1.2 iOS

- A default value can be defined
- Multiple variations support
- Platform support lowered from iOS 6.0 and up to iOS 5.0 and up.

4.1.3 Android

- A default value can be defined
- Multiple variations support

4.2 Version 2.3

August 6th 2013

4.2.1 Dashboard

- Multiple experiments support
- Experiment status (draft, active, archived)
- Experiments info and report merged on the same page

4.2.2 iOS

- Better getVariation callback handling (works even at the first launch of the app)
- Less http requests (the registration with the server is now cached)
- Multiple experiments support

4.2.3 Android

- Less http requests (the registration with the server is now cached)
- Multiple experiments support

4.3 Version 2.2

July 28th 2013

First release of the new Arise platform (2.x). Versions 2.0 and 2.1 were never released to the public.

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*