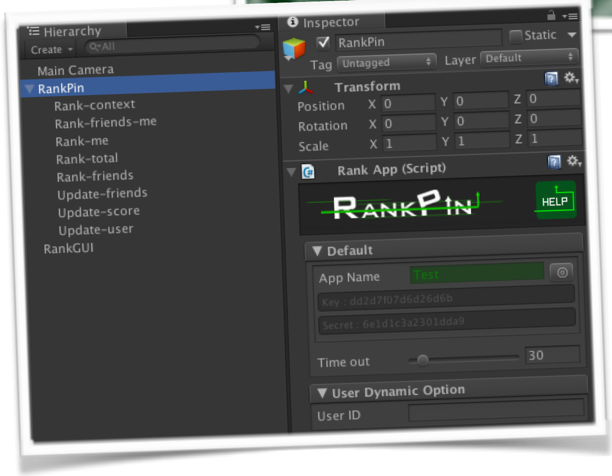
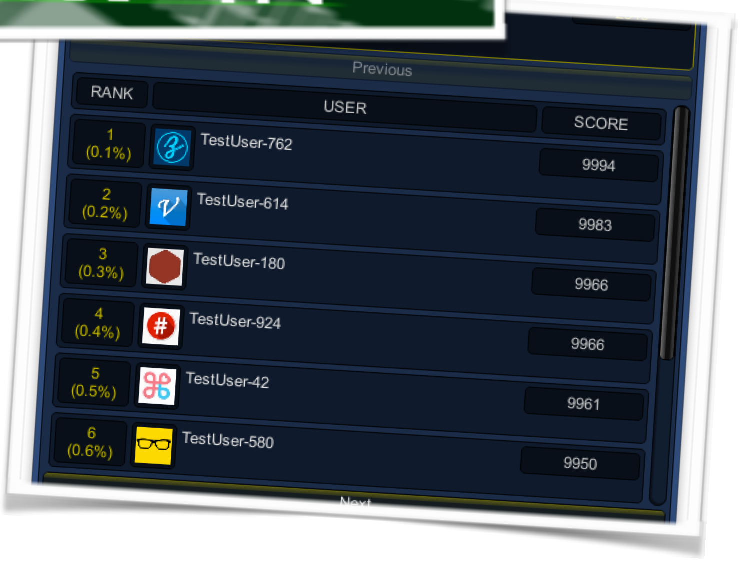
RankPin

Unity Ranking Service



```
Main Camera
▼ RankPin
    Rank-context
    Rank-friends-me
    Rank-me
    Rank-total
    Rank-friends
    Update-friends
    Update-score
    Update-user
  RankGUI
```

```
[✓] RankPin                         [ ] Static
Tag Untagged        Layer Default
▼ Transform
  Position   X 0   Y 0   Z 0
  Rotation   X 0   Y 0   Z 0
  Scale      X 1   Y 1   Z 1
▼ Rank App (Script)
  ▼ Default
    App Name   Test
    Key : dd2d7f07d6d26d6b
    Secret : 6e1d1c3a2301dda9
    Time out   30
  ▼ User Dynamic Option
    User ID
```



RANK	USER	SCORE
1 (0.1%)	TestUser-762	9994
2 (0.2%)	TestUser-614	9983
3 (0.3%)	TestUser-180	9966
4 (0.4%)	TestUser-924	9966
5 (0.5%)	TestUser-42	9961
6 (0.6%)	TestUser-580	9950

RankPin

Unity Ranking Service

Introduction

RankPin is designed to easily apply a ranking system with a simple setting in Unity. It provides a cost efficient method to simplify and speed up the process of Unity application and game development process. With a few clicks you will be able to apply a game ranking system.

Functions of RankPin

There are two functions of RankPin. First is to save data onto the server and the second to retrieve the saved data.

Saving data to server

- User data: Information (e.g. name, image url) required in Ranking UI
- Score: Accumulated score used for ranking.
- Friend list: Upload self and friend's list data.

Retrieving ranking data from server

- My ranking: My ranking from all users, my ranking between friends.
- All ranking: Accumulated ranking for all users, weekly all user ranking.
- Friend ranking: Accumulated all users ranking, weekly friend ranking.

* Weekly ranking is refreshed on Monday's.

Getting Started

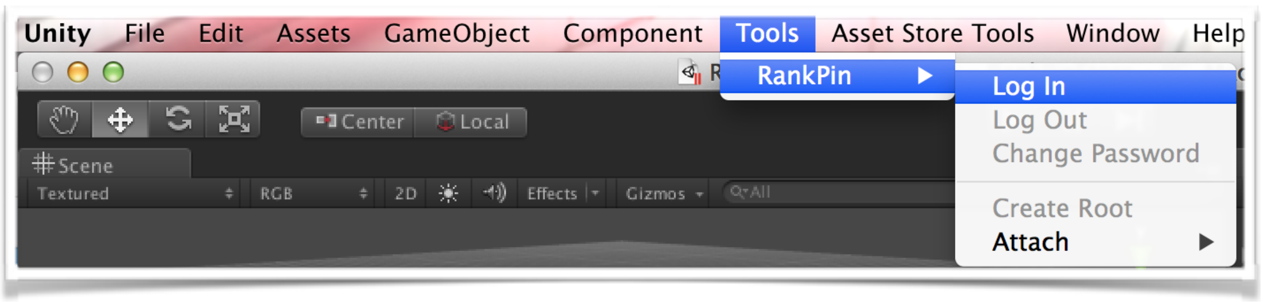
1. Log In / New Account : Login or create account from ranking server
2. Create Application : Create application in ranking server
3. Create Root and Application Setting : Create GameObject to be used in update and ranking
4. Update Setting : User data, score, friend's list update setting
5. Retrieve Ranking : My ranking, friend ranking, all ranking, weekly ranking, etc.

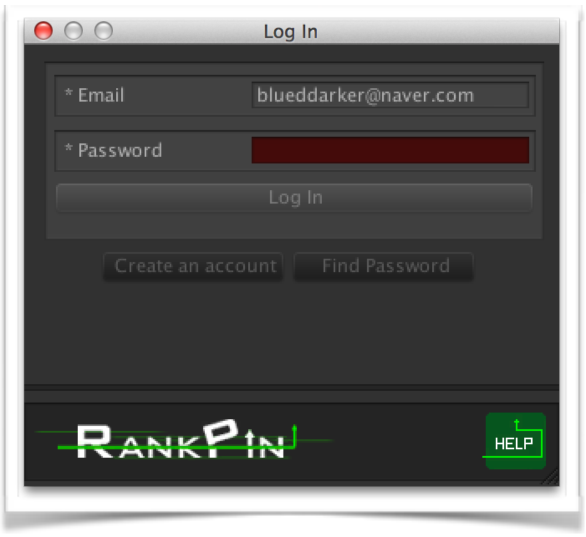
1. Log In / New Account

Create account or login with email address and password. You can create multiple applications with one account.

RankPin menu appears as below after selecting the Log In option.

Login




Log In

* Email blueddarker@naver.com

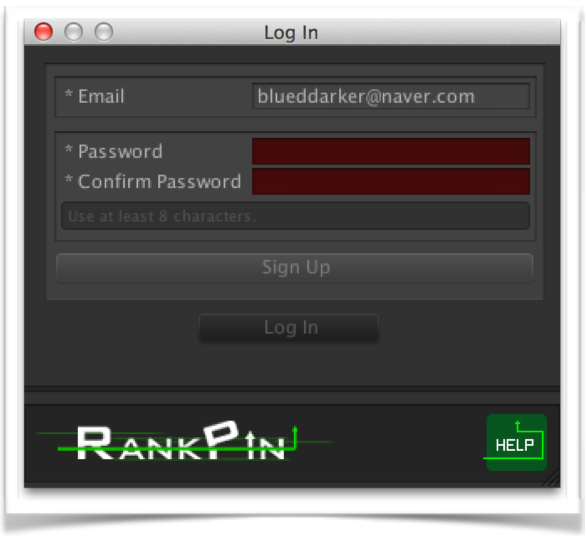
* Password

* Password : Combination of 8 or more characters and numbers

* Find Password : Used to find password

(Input account email address used in login)

Create Account


Log In

* Email blueddarker@naver.com

* Password

* Confirm Password

Use at least 8 characters.

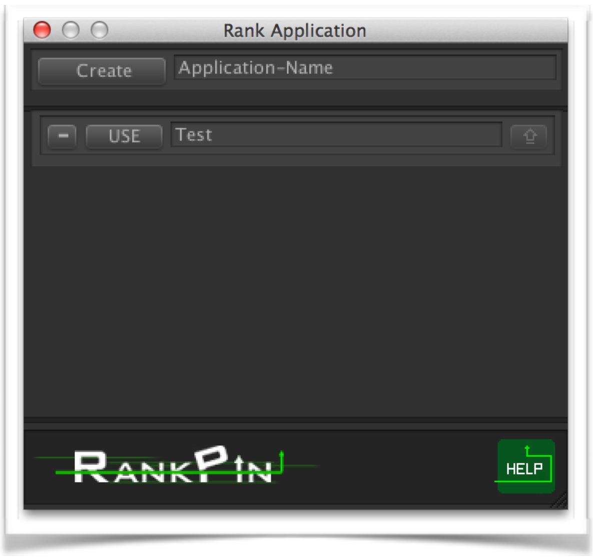
* Email : Active email address

(Email address must be active in order to use the find password option at a later date)

2. Create Application

Application must be created to apply ranking. You can create multiple applications with one account.

Below is the create application screen.



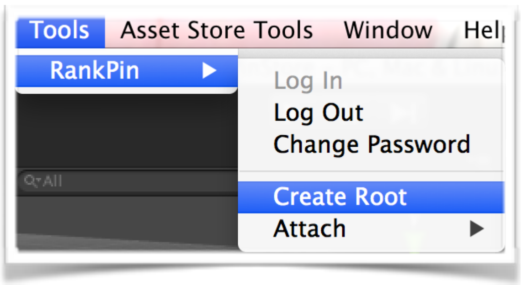
* [-] : Delete application. (All data on server will be deleted)

* USE : Apply current application.

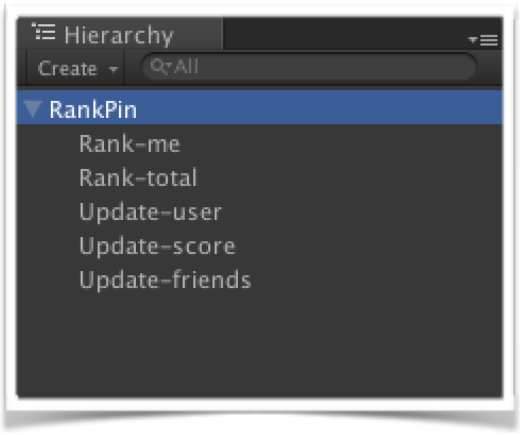
* Change name of application.

3. Create Root and Application Setting

Create Rank Root to use ranking.



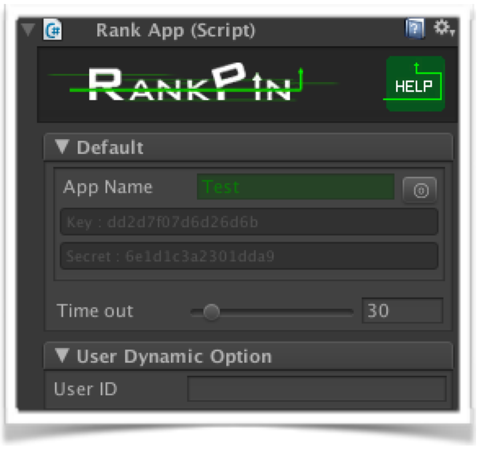
Select Create Root from RankPin menu.



GameObject is automatically created in the hierarchy.

- RankApp

Manage application setting and control data update and ranking data of Root GameObject.


Rank App (Script)

▼ Default
App Name: Test
Key : dd2d7f07d6d26d6b
Secret : 6e1d1c3a2301dda9
Time out: 30
▼ User Dynamic Option
User ID:

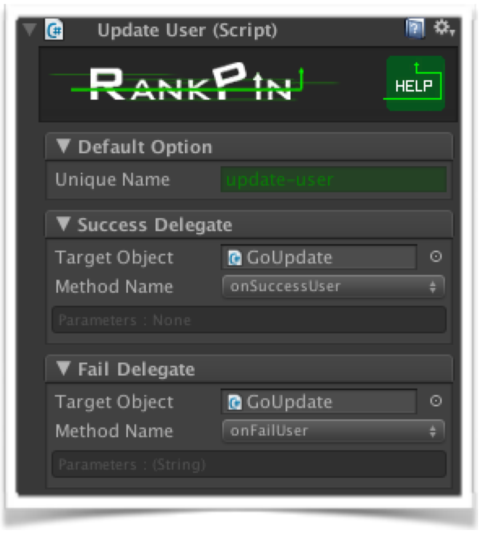
* Set App Name : Select application

* Time out : Error when there is no reply from server for a specific time

* User Dynamic Option : 소Set User ID in source code. (debug information)

- Update user

GameObject used to update my information.


Update User (Script)

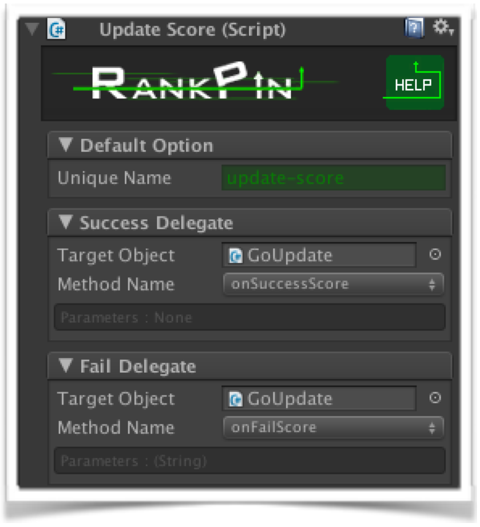
▼ Default Option
Unique Name: update-user
▼ Success Delegate
Target Object: GoUpdate
Method Name: onSuccessUser
Parameters : None
▼ Fail Delegate
Target Object: GoUpdate
Method Name: onFailUser
Parameters : (String)

* Success Delegate : Set to receive success information of server transmission.

* Fail Delegate : Set to receive fail information of server transmission.

- Update score

GameObject used to update my score information.



Update Score (Script)

▼ Default Option

Unique Name: update-score

▼ Success Delegate

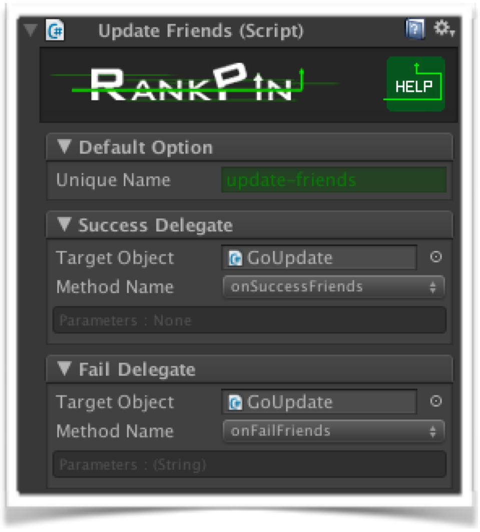
Target Object: GoUpdate
Method Name: onSuccessScore
Parameters : None

▼ Fail Delegate

Target Object: GoUpdate
Method Name: onFailScore
Parameters : (String)

- Update friends

GameObject used to update friend's list information.



Update Friends (Script)

▼ Default Option

Unique Name: update-friends

▼ Success Delegate

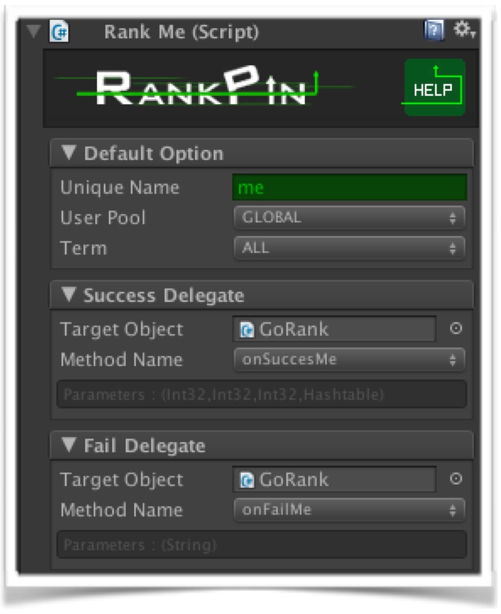
Target Object: GoUpdate
Method Name: onSuccessFriends
Parameters : None

▼ Fail Delegate

Target Object: GoUpdate
Method Name: onFailFriends
Parameters : (String)

- Rank me

GameObject used to perform retrieving operation of my ranking information.


Rank Me (Script)

HELP

▼ Default Option

Unique Name: me
User Pool: GLOBAL
Term: ALL

▼ Success Delegate

Target Object: GoRank
Method Name: onSuccesMe
Parameters : (Int32,Int32,Int32,Hashtable)

▼ Fail Delegate

Target Object: GoRank
Method Name: onFailMe
Parameters : (String)

* User Pool

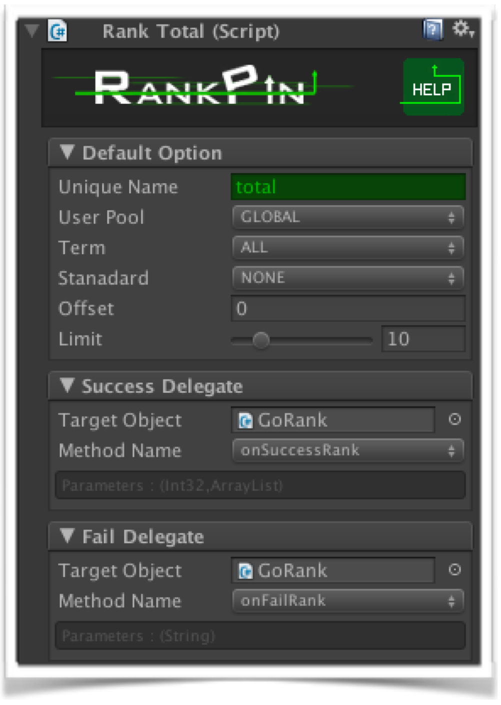
- GLOBAL : My ranking from all users
- FRIENDS : My ranking between friends

* Term

- ALL : Accumulated all ranking
- WEEK : Weekly ranking

- Rank total

GameObject used to perform retrieving operation of ranking list.


Rank Total (Script)

HELP

▼ Default Option

Unique Name: total
User Pool: GLOBAL
Term: ALL
Stanadard: NONE
Offset: 0
Limit: 10

▼ Success Delegate

Target Object: GoRank
Method Name: onSuccessRank
Parameters : (Int32,ArrayList)

▼ Fail Delegate

Target Object: GoRank
Method Name: onFailRank
Parameters : (String)

* Standard

- NONE : Retrieve ranking list from offset to limit
- ME : Retrieve ranking list above and below my score

* Offset : Ranking offset

* Limit : Number of user's ranking data

4. Update Setting

Perform operation of updating my information on server.

- User Data Update

Update my information on server. Unique ID is required to update.

Alternatively, you can upload various information in a Hashtable format for user data.

(Internally Hashtable data is converted into JSON format and size over 500 bytes will not be saved on server)

* Sample(Reference RankAppUpdate.cs)

Request

```
private void updateUser()
{
      Hashtable data = new Hashtable();
      data.Add("name", "Daniel");
      data.Add("url", “http://xxx.png“);
      this.updateUser(data);
}
```

Response

```
public override void onSuccessUser()
{
      base.onSuccessUser();
      Debug.Log("Update user informations.!");
}
public override void onFailUser(string message)
{
      base.onFailUser(message);
      Debug.LogWarning(message);
}
```

- Score Update

Perform operation of updating your ranking score information onto server.

Scores are all ranking, weekly ranking and friend ranking data.

(Score data uploads accumulated scores. Weekly rankings automatically determines current week’s score from server)

* Sample(Reference RankAppUpdate.cs)

Request

```
private void updateScore()
{
       int score = 102345;
       this.updateScore((uint)score);
}
```

Response

```
public override void onSuccessScore()
{
       base.onSuccessScore();
       Debug.Log("Update score.!");
}
public override void onFailScore(string message)
{
       base.onFailScore(message);
       Debug.LogWarning("message);
}
```

- Update Friend List

Your friend's list must be uploaded to server in order to use friend ranking. If you are not going to use this feature, the list does not need to be uploaded.

Friend's list must update User ID of friend's list to server in ArrayList.

* Sample(Reference RankAppUpdate.cs)

Request

```
private void updateFriends()
{
       ArrayList friends = new ArrayList();
       friends.Add("33");
       friends.Add("367");
       friends.Add("369");
       this.updateFriends(friends);
}
```

Response

```
public override void onSuccessFriends()
{
       base.onSuccessFriends();
       Debug.Log("Update friends list.!");
}
public override void onFailFriends(string message)
{
       base.onFailFriends(message);
       Debug.LogWarning(message);
}
```

5. Retrieve Ranking

Perform operation of retrieving ranking information from server.

- My Ranking

Retrieves my ranking information. You can retrieve my ranking from all, weekly and friend ranking.

* Sample(Reference RankAppRank.cs)

Request

```
private void requestRankMe()
{
	this.rankMe("me");
}
```

Response

```
public override void onSuccesMe(int total, int rank, int score, Hashtable data)
{
	base.onSuccesMe(total, rank, score, data);
	Debug.Log(string.Format("total:{0},rank:{1},score:{2}",total,rank,score));
}
public override void onFailMe(string message)
{
	base.onFailMe(message);
	Debug.LogWarning(message);
}
```

- All ranking

Retrieve ranking information from all users.

You can set offset and limit when retrieving ranking information.

- offset : Ranking offset
- limit : Numbers of user’s retrieved ranking data (maximum of 50)

* Sample(Reference RankAppRank.cs)

Request

```
private void requestRank()
{
	this.rank("total");
}
```

Response

```
public override void onSuccessRank(int total, ArrayList users)
{
	base.onSuccessRank(total, users);
	Debug.Log(string.Format("total:{0}", total));
	foreach(Hashtable user in users)
	{
		HashtableHelper.print("Rank", user);
		Hashtable data = (Hashtable)user[RankPin.RankConstants.KEY_DATA];
		if(data == null)
			continue;
		HashtableHelper.print("Data", data);
	}
}
public override void onFailRank(string message)
{
	base.onFailRank(message);
	Debug.LogWarning(message);
}
```

- Friend Ranking

Retrieve ranking information of friends.

* Sample(Reference RankAppSample.cs)

Request

```
private void requestRank()
{
	this.rank("friends");
}
```

Response

```
public override void onSuccessRank(int total, ArrayList users)
{
	base.onSuccessRank(total, users);
	Debug.Log(string.Format("total:{0}", total));
}
public override void onFailRank(string message)
{
	base.onFailRank(message);
	Debug.LogWarning(message);
}
```

- Weekly Ranking

Retrieve all and friend’s weekly ranking. Data is refreshed every Monday.

* Sample(Reference RankAppSample.cs)

Request

```
private void requestRank()
{
      this.rank("week");
}
```

Response

```
public override void onSuccessRank(int total, ArrayList users)
{
      base.onSuccessRank(total, users);
      Debug.Log(string.Format("total:{0}", total));
}
public override void onFailRank(string message)
{
      base.onFailRank(message);
      Debug.LogWarning(message);
}
```

Price

RankPin is free to use.

Samples

1. Simple

Example of applying ranking without RankApp.

- SimpleUpdate.cs : Sample code of updating user data, score and friend list
- SimpeRank.cs : Sample code of retrieving my ranking, all ranking and weekly ranking

2. RankApp_Simple

Example of applying ranking with RankApp. Create root through Create Root Menu.

- RankAppUpdate.cs : Sample code of updating user data, score and friend list
- RankAppRank.cs : Sample code of retrieving my ranking, all ranking and weekly ranking.

3. RankApp_GUI

Sample of outputting information to GUI with RankApp.

- RankAppSample.cs : Sample code of retrieving update and ranking information
- RankSampleGUI.cs : Sample code of outputting ranking information on screen