AppRegistry

Project with Native Code Required

If you are using the managed Expo workflow there is only ever one entry component registered with AppRegistry and it is handled automatically (or through registerRootComponent). You do not need to use this API.

AppRegistry is the JS entry point to running all React Native apps. App root components should register themselves with AppRegistry.registerComponent, then the native system can load the bundle for the app and then actually run the app when it's ready by invoking AppRegistry.runApplication.

To "stop" an application when a view should be destroyed, call AppRegistry.unmountApplicationComponentAtRootTag with the tag that was passed into runApplication. These should always be used as a pair.

AppRegistry should be required early in the require sequence to make sure the JS execution environment is setup before other modules are required.

Reference

Methods

getAppKeys()

```
static getAppKeys(): string[];
```

Returns an array of strings.

getRegistry()

```
static getRegistry(): {sections: string[]; runnables: Runnable[]};
```

Returns a Registry object.

getRunnable()

```
static getRunnable(appKey: string): : Runnable | undefined;
```

Returns a Runnable object.

Parameters:

| NAME | TYPE |
|-----------------|--------|
| appKey Required | string |

getSectionKeys()

```
static getSectionKeys(): string[];
```

Returns an array of strings.

getSections()

```
static getSections(): Record<string, Runnable>;
```

Returns a Runnables object.

registerCancellableHeadlessTask()

```
static registerCancellableHeadlessTask(
  taskKey: string,
  taskProvider: TaskProvider,
  taskCancelProvider: TaskCancelProvider,
);
```

Register a headless task which can be cancelled. A headless task is a bit of code that runs without a UI.

Parameters:

| NAME | TYPE | DESCRIPTION |
|-----------------------------|--------------------|---|
| taskKey Required | string | The native id for this task instance that was used when startHeadlessTask was called. |
| taskProvider Required | TaskProvider | A promise returning function that takes some data passed from the native side as the only argument. When the promise is resolved or rejected the native side is notified of this event and it may decide to destroy the JS context. |
| taskCancelProvider Required | TaskCancelProvider | a void returning function that takes no arguments; when a cancellation is requested, the function being executed by taskProvider should wrap up and return ASAP. |

registerComponent()

```
static registerComponent(
  appKey: string,
  getComponentFunc: ComponentProvider,
  section?: boolean,
): string;
```

Parameters:

| NAME | ТҮРЕ |
|----------------------------|-------------------|
| appKey Required | string |
| componentProvider Required | ComponentProvider |
| section | boolean |

registerConfig()

```
static registerConfig(config: AppConfig[]);
```

Parameters:

| NAME | TYPE |
|-----------------|-------------|
| config Required | AppConfig[] |

registerHeadlessTask()

```
static registerHeadlessTask(
  taskKey: string,
  taskProvider: TaskProvider,
);
```

Register a headless task. A headless task is a bit of code that runs without a UI.

This is a way to run tasks in JavaScript while your app is in the background. It can be used, for example, to sync fresh data, handle push notifications, or play music.

Parameters:

| NAME | TYPE | DESCRIPTION |
|--------------------------|--------------|---|
| taskKey Required | string | The native id for this task instance that was used when startHeadlessTask was called. |
| taskProvider Required | TaskProvider | A promise returning function that takes some data passed from the native side as the only argument. When the promise is resolved or rejected the native side is notified of this event and it may decide to destroy the JS context. |

registerRunnable()

```
static registerRunnable(appKey: string, func: Runnable): string;
```

Parameters:

| NAME | TYPE |
|-----------------|----------|
| appKey Required | string |
| run Required | function |

registerSection()

```
static registerSection(
  appKey: string,
  component: ComponentProvider,
);
```

Parameters:

| NAME | TYPE |
|--------------------|-------------------|
| appKey Required | string |
| component Required | ComponentProvider |

runApplication()

```
static runApplication(appKey: string, appParameters: any): void;
```

Loads the JavaScript bundle and runs the app.

Parameters:

| NAME | ТҮРЕ |
|------------------------|--------|
| appKey Required | string |
| appParameters Required | any |

setComponentProviderInstrumentationHook()

```
static setComponentProviderInstrumentationHook(
  hook: ComponentProviderInstrumentationHook,
);
```

Parameters:

| NAME | TYPE |
|---------------|----------|
| hook Required | function |

A valid hook function accepts the following as arguments:

| NAME | TYPE |
|----------------------------------|--------------------|
| component Required | ComponentProvider |
| scopedPerformanceLogger Required | IPerformanceLogger |

The function must also return a React Component.

setWrapperComponentProvider()

```
static setWrapperComponentProvider(
  provider: WrapperComponentProvider,
);
```

Parameters:

| NAME | ТҮРЕ |
|-------------------|-------------------|
| provider Required | ComponentProvider |

startHeadlessTask()

```
static startHeadlessTask(
  taskId: number,
  taskKey: string,
  data: any,
);
```

Only called from native code. Starts a headless task.

Parameters:

| NAME | TYPE | DESCRIPTION |
|-----------------|--------|--|
| taskId Required | number | The native id for this task instance to keep track of its execution. |

| NAME | TYPE | DESCRIPTION |
|---------------------|--------|--------------------------------|
| taskKey Required | string | The key for the task to start. |
| data Required | any | The data to pass to the task. |

unmountApplicationComponentAtRootTag()

```
static unmountApplicationComponentAtRootTag(rootTag: number);
```

Stops an application when a view should be destroyed.

Parameters:

| NAME | ТҮРЕ |
|------------------|--------|
| rootTag Required | number |

Type Definitions

AppConfig

Application configuration for the registerConfig method.

| TYPE | |
|--------|--|
| object | |

Properties:

| NAME | ТҮРЕ |
|-----------------|-------------------|
| appKey Required | string |
| component | ComponentProvider |

| NAME | ТҮРЕ |
|---------|----------|
| run | function |
| section | boolean |

Note: Every config is expected to set either component or run function.

Registry

| TYPE | |
|--------|--|
| object | |

Properties:

| NAME | ТҮРЕ |
|-----------|--------------------|
| runnables | array of Runnables |
| sections | array of strings |

Runnable

| TYPE | |
|--------|--|
| object | |

Properties:

| NAME | ТҮРЕ |
|-----------|-------------------|
| component | ComponentProvider |
| run | function |

Runnables

An object with key of appKey and value of type of Runnable.

| TYPE | |
|--------|--|
| object | |

Task

A Task is a function that accepts any data as argument and returns a Promise that resolves to undefined.

TYPE function

TaskCanceller

A TaskCanceller is a function that accepts no argument and returns void.

TYPE function

TaskCancelProvider

A valid TaskCancelProvider is a function that returns a TaskCanceller .

TYPE function

TaskProvider

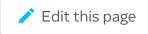
A valid TaskProvider is a function that returns a Task.

TYPE function

Is this page useful?







Last updated on **Jun 21, 2023**