iOS

1, set `autoresizingMask` in `CCEAGLView-ios.mm`, otherwise, the callback `didRotateFromInterfaceOrientation` would not be called.

```
- (id) initWithFrame:...
    self.autoresizingMask = UIViewAutoresizingFlexibleHeight|UIViewAutoresizingFlexibleWidth;
```

2, define `setOrientation(bool land)` in `RootViewController.mm`

```
bool iosPortrait = false;
bool iosInited = false;

- (void)viewDidLoad {
    [super viewDidLoad];

    iosInited = true;
}

void setOrientationiOS(bool land){
    iosPortrait = !land;
    [UIView setAnimationsEnabled: false];
    NSNumber *value = [NSNumber numberWithInt: land ? UIInterfaceOrientationLandscapeRight :
UIInterfaceOrientationPortrait];
    [[UIDevice currentDevice] setValue:value forKey:@"orientation"];
    [UIViewController attemptRotationToDeviceOrientation];
    [UINavigationController attemptRotationToDeviceOrientation];
    [UIView setAnimationsEnabled: true];
}
```

3, define `supportedInterfaceOrientations` and `shouldAutorotate` in `RootViewController.mm`, otherwise, the screen orientation can not be changed.

```
#ifdef __IPHONE_6_0
- (NSUInteger) supportedInterfaceOrientations{
    return iosPortrait ? UIInterfaceOrientationMaskPortrait : UIInterfaceOrientationMaskLandscape;
}
#endif

- (BOOL) shouldAutorotate {
    return YES;
}
```

4, define `supportedInterfaceOrientationsForWindow` in `AppController.mm`, thus we can freely define the launch orientation in `TARGETS/General/Device Orientation`. Otherwise, we must check all launch orientations in `TARGETS/General/Device Orientation` and the launch screen would be strange in some iPhone set, and some ad would be crashed because we not defined the desired orientation.

```
- (NSUInteger)application:(UIApplication *)application supportedInterfaceOrientationsForWindow:(UIWindow
*)window
{
    extern bool iosInited;
    return iosInited ? UIInterfaceOrientationMaskAllButUpsideDown : UIInterfaceOrientationMaskLandscape;
    // we use landscape if not inited because our game is primarily displayed in landscape
}
```

5, set desired launch orientation in `TARGETS/General/Device Orientation`, if the app like landscape, then we can set Landscape Left and Landscape Right only

6, set iOS deployment Target to 9.0 or above in `PROJECT/Info/Deployment Target`

Android

1, we can define `setOrientationAndroid(bool land)` in any c++ file

```
void setOrientationAndroid(bool landscape){
#if CC_TARGET_PLATFORM == CC_PLATFORM_ANDROID
    JniMethodInfo methodInfo;
    if (!JniHelper::getStaticMethodInfo(methodInfo, "org/cocos2dx/cpp/AppActivity", "setOrientationAndroid",
"(Z)V"))
    {
        CCLOG("%s %d: error to get methodInfo", __FILE__, __LINE__);
        return;
    }
    methodInfo.env->CallStaticVoidMethod(methodInfo.classID, methodInfo.methodID, landscape);
    methodInfo.env->DeleteLocalRef(methodInfo.classID);
#endif
}
```

2, we define the `setOrientationAndroid(bool land)` function in `AppActivity.java`

```
static AppActivity $this;

protected void onCreate(Bundle savedInstanceState) {
    ...
    $this = this;
    ...
}

public static void setOrientationAndroid(boolean landscape){
    if ($this == null){
        return;
    }

    $this.setRequestedOrientation(landscape ? ActivityInfo.SCREEN_ORIENTATION_SENSOR_LANDSCAPE :
ActivityInfo.SCREEN_ORIENTATION_SENSOR_PORTRAIT);
}
```

Cocos

1, override `virtual void applicationScreenSizeChanged(int newWidth, int newHeight);` in `AppDelegate.h` and `AppDelegate.cpp`

```
static bool isLand = true;
static void onScreenSizeChanged(GLView* glview, int newWidth, int newHeight){
    auto portrait = newHeight > newWidth;
    if (protrait == isLand){
        return;
    }
    glview->setFrameSize(newWidth, newHeight);

    if (portrait){
        float v_ = (float)newWidth / (float)newHeight;
        auto w = fmin(designSize.width, designSize.height);
        auto h = fmax(designSize.width, designSize.height);
        if (v_ < 0.6f) {
          glview->setDesignResolutionSize(w, h, ResolutionPolicy::FIXED_WIDTH);
        } else {
          glview->setDesignResolutionSize(w, h, ResolutionPolicy::FIXED_HEIGHT);
        }
```

```
    } else {
        float v_ = (float)newHeight / (float)newWidth;
        auto w = fmax(designSize.width, designSize.height);
        auto h = fmin(designSize.width, designSize.height);
        if (v_ > 0.6f) {
          glview->setDesignResolutionSize(w, h, ResolutionPolicy::FIXED_WIDTH);
        } else {
          glview->setDesignResolutionSize(w, h, ResolutionPolicy::FIXED_HEIGHT);
        }
    }

    Director::getInstance()->getEventDispatcher()->dispatchCustomEvent("glview_window_resized", nullptr);
}

void AppDelegate::applicationScreenSizeChanged(int newWidth, int newHeight){
    auto director = Director::getInstance();
    auto glview = director->getOpenGLView();
    if (glview)
    {
        onScreenSizeChanged(glview, newWidth, newHeight);
    }
}
```

2, define `setScreenOrientation` in `AppDelegate.cpp`

```
void setScreenOrientation(bool land){
    isLand = land;
#ifdef CC_PLATFORM_PC
    auto director = Director::getInstance();
    auto glview = director->getOpenGLView();
    if (glview != nullptr){
        auto size = glview->getFrameSize();
        if (land){
            onScreenSizeChanged(glview, fmax(size.width, size.height), fmin(size.width, size.height));
        } else {
            onScreenSizeChanged(glview, fmin(size.width, size.height), fmax(size.width, size.height));
        }
    }
#elif CC_TARGET_PLATFORM == CC_PLATFORM_IOS
    extern void setOrientationiOS(bool land);
    setOrientationiOS(land);
#elif CC_TARGET_PLATFORM == CC_PLATFORM_ANDROID
    extern void setOrientationAndroid(bool land);
    setOrientationAndroid(land);
#endif
}
```

3, if we use fairygui, we would register `glview_window_resized` event

```
bool GRoot::initWithScene(cocos2d::Scene* scene, int zOrder){
    ...
    _windowSizeListener = Director::getInstance()->getEventDispatcher()-
>addCustomEventListener("glview_window_resized", CC_CALLBACK_0(GRoot::onWindowSizeChanged, this));
    ...
}
```

Usage

```
setScreenOrientation(true); // landscape
setScreenOrientation(false); // portrait
```