Welcome to Paintcraft!

Thanks so much for the interest in this package – we think you're really going to like using Paintcraft to build and launch your high-quality drawing applications.

We are doing our best to make this package artist friendly and really flexible so it will satisfy any of your needs. This quick tutorial is intended to walk anyone familiar with Unity3D through the basics of Paintcraft package in just about *15-30 minutes*. Hopefully you'll learn how to make drawing application as fast as possible.

Package Structure

```
PaintCraft
 ├─ 3DParty
 │   └─ NodeInspector          # (3) OpenSource graph editor used in brush configuration
 ├─ BrushPack                  # (1) Contains built-in tools (ink, Cryon, etc)
 │   ├─ Advanced               # available in both (Base and Pro version)
 │   └─ Basic
 ├─ Demo
 │   ├─ ColoringBook           # (5) Complete coloring book project (fill free to use it for rockstring)
 │   ├─ DrawingObjects         # Moving by animation or by script drawing game objects
 │   ├─ DrawOnExistingImage    # Drawing on top of existing image
 │   │                           (eraser area is to the original image rather than canvas color)
 │   ├─ OnePlayerBasic         # Simple scene for one player coloring book
 │   ├─ PatternBrush           # Scene where you can use pattern brush
 │   ├─ Prefabs                # Prefabs used by Demo scenes
 │   ├─ StampBrush             # Here you can use stamps(stickers)
 │   ├─ SwitchPageLive         # Demonstrate that you can switch pages without actual reloading scene
 │   ├─ TransparentBackground  # Demonstrate coloring book with transparent background
 │   ├─ TwoPlayersOneCanvas    # Split screen mode where each player draw on the same coloring page
 │   └─ TwoPlayersTwoCanvases  # Split screen mode where player draw own image
 ├─ Engine                     # (4) Full source code of paintcraft package
 ├─ Prefabs                    # Paintcraft prefab gameobject
 └─ Pro                        # (2) This folder available only for customers purchased pro package
     ├─ 3D                     # 3D scenes
     └─ Svg                    # SVG packages
         ├─ SimplySVG          # Import this package for SimplySVG integration
         └─ SVGImporter        # Import this package for SVGImporter integration
```

1. Brush configuration is really flexible. You can controll your line almost at any point. And this plugin used to make it more user friendly. here is an example of one of the brush config
2. We included several brushes to this package so you can freely use them in your project. and for sure you can play with any parameters.
3. This is a live project with picture selection and 2 players split screen drawing. You can test it here in the browser or on your mobile WindowsStore, Android
4. This is a source code of the package. if you are a programmer maybe it will be interesting to check the source code. It is available in base and pro package.
5. This folder available only for customers purchased PRO package and it contains SVG integration and 3D drawing implementation. If you are doing VR app probably it's what you need.

PaintCraft prefab

To start your first Drawing app project, the easiest thing to do is to use the `PaintCraft prefab` which is located at `Assets/Paintcraft/Prefab` folder.

Find this object in your project and drag it to the scene

If you have a brand new scene you will get this structure

This object already contains necessary configs, but it won't work immediately because we need to tell our canvas where we want to draw. In case if you have `DirectionalLight` object just delete it. It comes from 3D default scene setup

If you try to scene you will see the error in the console

As you see it say that you have to provide page config. to fix that you need to Select PaintcraftCanvas game object and select page config. But first of all let's create new page config

Page Config {#page-config}

Let's create a new Page Config. This object will keep configuration just the size our canvas will be.

1. Create a new folder for this `Assets/Configs`
2. Click right mouse button and select `Create/PaintCraft/BlankPageConfig` menu`![](/assets/create blank page config.png)`
3. Setup options at your page config | Name | Description | | :--- | :--- | | UniqueId | This settings must be unique across all pages inside the project. This id is used to store icons or actual changes so if you open your page next time you will see previous changes | | StartImageBackground | Background image which will draw on top of | | Size | Actual size of the image in pixels (like 800x600). You are free to use any sizes and this size does not relate to the screen size as you can zoom your image and draw more details |

After you set your configs you will get something like this

`![](/assets/blank page config inspector.png)`

Apply new page config {#apply-new-page-config}

Now you have a page config and you can use it with your canvas object

1. Select PaintcraftCanvas object in the hierarchy `![](/assets/select paintcraft canvas object in hierarchy.png)`
2. Link your page config to the `Page Config` property in the same `Canvas Controller` component change `![](/assets/set page config at canvas controller.png)`
3. You can also change default background color of your canvas `Default BG Color`. I'll use default `white` color

Here you go. Now press play button and you will be able to draw on your screen

`![](/assets/drawing smile.png)`

Change Tools Button

Now we have a simple canvas. let's create some simple UI controls to change our tools.

Ink Button {#ink-button}

As you can see we have `Basic` and `Advanced` brush packs in the `Assets/PaintCraft/BrushPack` folder. And you can freely use them in your projects.

Let's create standard unity buttons which will change some tools

1. Create one UI button `GameObject/UI/Button` I'll put it to the left bottom corner and name it `Ink`
2. Add `ChangeBrushOnClickController` on to it
3. Now you need to set 2 properties on this component so result will looks like this
4. Set `LineConfig` refference to our prefab subcomponent on our scene `PaintCraft/Canvas/LineConfig`. Actually you can click right circle on the right side of this property and it will show you just one possible object, because we have only one right now.
5. And final step is to set link to our tools. use the same circle and select `Ink` config from our basic pack. Don't forget to switch to `Assets` tab because this is a config stored in our project and not a scene object

Now you can start your scene and try to draw with the brush (because it's already selected in our line config as default tool) and after you click `Ink` button you will draw with ink

Spray Button {#spray-button}

If you want to add more tools just clone InkButton object and change refference to the tool

1. To clone select select `Ink Button` gameobject in the Hierarchy panel and then click `Edit/Duplicate` or just `ctrl+D` or `cmd+D` on mac
2. Rename new object to `Spray Button`
3. Change text on the button to `Spray`
4. Change position of this object. I'll change PosY in the rect transform so it will be alligned with my `Ink` button
5. Setup Link to spray at `ChangeBrushOnClickController`

And now let's change our default tool in our line config. Select `PaintCraft/LineConfig` object. Click on `Brush` and select `Ink`. You can also change default scale at LineConfig so your line will be thinner.

PaintCraft Controls Button

At this step we will create 3 buttons `Undo` `Redo` and `Clear`

Clear Button {#clear-canvas-button}

1. As usuall let's create button using top menu `GameObject/UI/Button` and put it to the right bottom control. To do that you can select RectTransform and change pivot and position at once. You need to click on the left top rect then hold down shift+alt click the bottom right corner
2. Set small offset. set `PosX=-10 PosY=10` so it won't be right on the corner and change `Width=120`.
3. Name this button as `Clear`
4. Now we need to ad event to this button. Click `+` in `OnClick()` event
5. Click on the object field and select `Canvas` object to this property. You can also drag and drop `PaintCraft/Canvas` scene object to this property
6. You need to select the `Canvas.ClearCanvas()` function. To do that click on `No Function` button select `CanvasController` and `ClearCanvas` function.

Thats it. start your scene, try to draw something and then clear canvas

Undo/Redo {#undo/redo}

Let's repeat button creation process for undo/redo

1. Clone your `Clear` button 2 times.
2. Setup `(1)` cloned button position. change caption, name and set following position: `PosX=-75 PosY=50 Width=55`
3. Setup `(2)` cloned button position. change caption, name and set following position: `PosX=-10 PosY=50 Width=55`. You should get nicely alligned buttons at the right bottom corner, something like this:
4. Select `Undo` button and Remove `OnClick` handler using `minus` button
5. Add `UndoActiveButtonController` to `UndoButton`
6. Select `Canvas` for this controller you will get this. Make sure that `OnClick()` is cleared properly
7. Select `Redo` and repeat step 4 to 6 but this time add `RedoActiveButtonController`

Your systems buttons are ready and you can start your scene and check that everything is works properly. You can even see that undo/redo is clickable only if you have something to undo or redo. By default history size is 10.

Line Color and Thickness

Color {#color}

Let's create several buttons which will shift our colors. First we need to create panel.

1. Select unity UI `Canvas` and create a new child
2. Let's place it at the middle of our screen
3. Name it `ColorsPalette` and set `PosX=10` and `height=30`
4. Add standard unity ui `HorizontalLayoutGroup` script on it set `spacing=10` and `childForceExpand=false`. Then Add `ContentSizeFitter` and set `HorizontalFit=MinSize`

Now our root objects for our colors is ready and it would have nice feature. it would grow automatically and you can add several colors and all of them would be automactically aligned. so let's do that

1. create `GameObject/UI/Button` and make it as child for our `ColorsPalette` element and name it 'ColorButton'
2. Remove text `Text` child object of our button. so you won't have any caption on it
3. Set button size `Width=30` and `Height=30`
4. Now you can clone our buttons as many times as you want to have in your final app. I'll make 5 colors. here is hierarchy and how it looks on scene
5. Now we need to change color of our button. Select `ColorButton` object `ImageComponent` and click on `Color` property and in dropdown list select required color. Repeat this step for each `ColorButton`
6. Final step would be add color change controller to do that you need to select all color buttons buttons in inspector

   then add `ChangeColorOnClickController` And finaly copy our colors from our Image component to this controller. To Do that you need to select component menu (little gear at the top right corner of our component) and select `Copy Color from Image`

   and then select `LineConfig` ad you did that in tool change step.Now check all your color buttons and make sure that all of them has link to `LineConfig` and every one has the same color as button.

Click play button and try to change line of your color

Coloring Page

Drawing on plane canvas it's cool but let me show you how to create a coloring page. where all your drawings would be restricted to predefined regions. We would need to setup 3 image and i'll explain you where each of them is used for.

Page Config {#page-conifg}

1. First of all we need to create a new page config go to your `Configs` folder and create new page config file. but this time select `RightClickButton/Create/PaintCraft/ColoringPageConfig` .
2. As usual you need to setup `UniqueId` property. `Start image` if you want to use it and setup 3 image links. Right now let'd just define `UniqueId`
3. Here i should notice you that every image used as coloring page must be stored somwhere in `Resources` folder so it won't be loaded automaticaly to the memory and your app will works on any mobile device even if you will have on handreed pages. So let's create Resources folder first and one subfolder for our image `NewPage1` and place our images(icon, outline and region) there.

OutlineLayer {#outlinelayer}

In coloring page we use 2 layers. One layer is the same as on blank page where we draw our graphics. and 2 layer contains our actual image. On the screenshot below we hasn't drawn anything and our back layer is blank and white. Outline layer contain Bear image.

And as you see Outline layer must have transparency. you could have any colors on it. something like shades color injections and so on.it's not mandatory to have black and white image. You can open demo project pictures and see that it has some transparent shades like on the png image below (shadows below the bird and on wings)

Then you need to fix import settings. By default unity could use power of 2 settings so your image would be stretched. to fix that select your outline image and then in inspector change `TextureType=Advanced (Default on Unity5 and later)` and then `Non Power of 2 = None` and also we don't need MipMaps so you could disable them as well. Here you could check what i have.

Icon {#icon}

We need separated downscaled icon because of the following reason. When you make any changes on the canvas we store this state on the disk. so when your player would comeback it will be able to work with this image. And we do not store mix of the backgraound layer (where you draw) and outline layer and it's just a lines which you made. The stored layer contains just user changes and looks like this (after result of floodfill with black and green color)

And we need to blend this with outline layer. It won't work well on older devices if we will blend this with original outline layer. because each image would consume to much memory. and this is why we need to add downscaled image.

Icon size depends on you which size of image do you want to have. if you will open canvas controller you will see the settings for the icon size by default it `440x330`

For import settings you could have the same import settings which you used for outline layer

Predefined Regions {#predefined-regions}

To make region texture what you need to do is just paint your texture in image editor. where every region has own unique color. And if you want to merge some regions you can assign the same color to it. For example you can make the same color for all grass on the image. or for all body parts so if player click on one region with bucket all body parts will be filled with the same region.

- Any _regions where alpha = 0 will be ignored _for painting. This means that if you don't want to allow user paint some regions just assign alpha = 0 to it.
- Finally you need to make proper setting for the texture. Critical part here is a **FilterMode **it must have **"Point (no filter)" **value

Final config {#final-config}

At this point you should have all 3 images. let's setup links to them in our config file

If your images would be stored not inside Resource folder. this inspector will notify you about this. Also if you will change location of this images you would nedd to setup this link again. unity won't change lnks to them automatically as it does for scene or prefab objects.

Now you could select this config in your `CanvasController` component.

And now you could start your scene and start drawing your image