# Mobile Utils

This extension allows **Android** and **iOS** developers to interact with the camera and gallery applications of the mobile device.

> INFO In order to avoid injecting unnecessary permissions into the application manifest this extension is separate into four distinct modules depending on your game requirements.

## Camera Module

These functions are provided to work with the camera.

- **MobileUtils_Camera_Open**

## Gallery Module

These functions are provided to work with the gallery.

- **MobileUtils_Gallery_Open**

## Image Tools Module

These functions are provided to work with images and edit them.

- **MobileUtils_Image_Crop**
- **MobileUtils_Image_Height**
- **MobileUtils_Image_Resize**
- **MobileUtils_Image_Width**

# Share Module

These functions are provided to share files.

- MobileUtils_Share_Open

# MobileUtils_Camera_Open

This functions opens the mobile device camera interface.

> NOTE On iOS devices, permissions will be requested automatically. Android developers already have the required permissions added to the manifest by the extension: `android.permission.READ_EXTERNAL_STORAGE` , `android.permission.WRITE_EXTERNAL_STORAGE` , `android.permission.CAMERA`

**Syntax:**

```
MobileUtils_Camera_Open();
```

**Returns:**

```
N/A
```

**Triggers:**

```
Asynchronous Social Event
```

| async_load Contents | | |
|---|---|---|
| **Key** | **Type** | **Description** |
| type | string | `"MobileUtils_Camera_Open"` |
| path | string | The path to the photo |

**Example:**

```
var read = "android.permission.READ_EXTERNAL_STORAGE"
var write = "android.permission.WRITE_EXTERNAL_STORAGE"
var camara = "android.permission.CAMERA"

if(os_type == os_ios or os_check_permission(write) and os_check_permission(read) and
os_check_permission(camara))
{
    MobileMedia_Camera_Open()
}
```

```
else
{
    if(!os_check_permission(write) and !os_check_permission(read))
        os_request_permission(write,read)

    if(!os_check_permission(camara))
        os_request_permission(camara)
}
```

The code sample above triggers the mobile device camera overlay that will allow users to take a photo. This photo path can be later caught inside an **Async Social** event.

```
if(async_load[?"type"] == "MobileMedia_Camera_Open")
{
    var path = async_load[?"path"]
    Obj_MobileMedia_Camera_Picture.sprite = sprite_add(path, 0, 0, 0, 0, 0);
}
```

The code above matches the response against the correct event **type** and loads the newly taken photo.

# MobileUtils_Gallery_Open

This functions opens the mobile device gallery interface.

**Syntax:**

```
MobileUtils_Gallery_Open()
```

**Returns:**

```
N/A
```

**Triggers:**

```
Asynchronous Social Event
```

| async_load Contents | | |
|---|---|---|
| **Key** | **Type** | **Description** |
| type | string | `"MobileUtils_Gallery_Open"` |
| path | string | The path to the image |

**Example:**

```
MobileUtils_Gallery_Open()
```

The code sample above triggers the mobile device gallery overlay, allowing the user to select a pohto, this will later trigger an Async Social event.

```
if(async_load[?"type"] == "MobileUtils_Gallery_Open")
{
    var path = async_load[?"path"]
    MobileUtils_Image_Resize(path,300,300)
    MobileUtils_Image_Crop(path,100,100,100,100)
    Obj_MobileUtils_Gallery_Picture.sprite = sprite_add(path,0,0,0,0,0)
}
```

The code above matches the response against the correct event **type** and proceeds to resize cropt and load the sprite into a variable.

# MobileUtils_Image_Crop

This function will crop the image in the specified path.

### Syntax:

```
MobileUtils_Image_Crop(path, w, h, x, y);
```

| Argument | Type | Description |
|---|---|---|
| path | string | The path to the image |
| w | real | The width of the zone to crop |
| h | real | The height of the zone to crop |
| x | real | The x offset of the zone to crop |
| y | real | The y offset of the zone to crop |

### Returns:

```
N/A
```

### Example:

```
if(async_load[?"type"] == "MobileUtils_Gallery_Open")
{
    var path = async_load[?"path"]
    if(MobileUtils_Image_Width(path) != 300 or MobileUtils_Image_Height(path) != 300)
    {
        MobileUtils_Image_Resize(path,300,300)
        MobileUtils_Image_Crop(path,100,100,100,100)
        MobileUtils_Image_Resize(path,300,300)
    }
    Obj_MobileMedia_Gallery_Picture.sprite = sprite_add(path,0,0,0,0,0)
}
```

The code above is was extracted from the **MobileMedia_Gallery_Open** sample and shows an example on how to crop an image (for a complete example please refere to the restective function).

# MobileUtils_Image_Height

This function returns the height of the image in the specified path.

### Syntax:

```
MobileUtils_Image_Height(path);
```

| Argument | Type | Description |
|---|---|---|
| path | string | The path to the image |

### Returns:

```
real
```

### Example:

```
if(async_load[?"type"] == "MobileUtils_Gallery_Open")
{
    var path = async_load[?"path"]
    if(MobileUtils_Image_Width(path) != 300 or MobileUtils_Image_Height(path) != 300)
    {
        MobileUtils_Image_Resize(path,300,300)
        MobileUtils_Image_Crop(path,100,100,100,100)
        MobileUtils_Image_Resize(path,300,300)
    }
    Obj_MobileMedia_Gallery_Picture.sprite = sprite_add(path,0,0,0,0,0)
}
```

The code above is was extracted from the **MobileMedia_Gallery_Open** sample and shows an example on how to get the height of an image (for a complete example please refere to the restective function).

# MobileUtils_Image_Resize

This function resizes the image in the path.

### Syntax:

```
MobileUtils_Image_Resize(path,w,h);
```

| Argument | Type | Description |
|---|---|---|
| path | string | The path to the image |
| w | real | The width of the zone to crop |
| h | real | The height of the zone to crop |

### Returns:

```
N/A
```

### Example:

```
if(async_load[?"type"] == "MobileUtils_Gallery_Open")
{
    var path = async_load[?"path"]
    if(MobileUtils_Image_Width(path) != 300 or MobileUtils_Image_Height(path) != 300)
    {
        MobileUtils_Image_Resize(path,300,300)
        MobileUtils_Image_Crop(path,100,100,100,100)
        MobileUtils_Image_Resize(path,300,300)
    }
    Obj_MobileMedia_Gallery_Picture.sprite = sprite_add(path,0,0,0,0,0)
}
```

The code above is was extracted from the MobileMedia_Gallery_Open sample and shows an example on how to resize an image (for a complete example please refere to the restective function).

# MobileUtils_Image_Width

This function returns the width of the image in the specified path.

### Syntax:

```
MobileUtils_Image_Resize(path);
```

| Argument | Description |
|---|---|
| path | The path to the image |

### Returns:

```
real
```

### Example:

```
if(async_load[?"type"] == "MobileUtils_Gallery_Open")
{
    var path = async_load[?"path"]
    if(MobileUtils_Image_Width(path) != 300 or MobileUtils_Image_Height(path) != 300)
    {
        MobileUtils_Image_Resize(path,300,300)
        MobileUtils_Image_Crop(path,100,100,100,100)
        MobileUtils_Image_Resize(path,300,300)
    }
    Obj_MobileMedia_Gallery_Picture.sprite = sprite_add(path,0,0,0,0,0)
}
```

The code above is was extracted from the **MobileMedia_Gallery_Open** sample and shows an example on to get the width of an image (for a complete example please refere to the restective function).

# MobileUtils_Share_Open

This functions opens the mobile device share interface.

> **NOTE** On iOS devices, permissions will be requested automatically. Android developers already have the required permissions added to the manifest by the extension: `android.permission.READ_EXTERNAL_STORAGE`, `android.permission.WRITE_EXTERNAL_STORAGE`, `android.permission.CAMERA`

**Syntax:**

```
MobileUtils_Share_Open(title, path, MIME)
```

| Argument | Type | Description |
|---|---|---|
| title | string | Title of the share (not available for all the providers) |
| path | string | The path to the file |
| MIME | string | One of the following strings: `"text/plain"`, `"text/rtf"`, `"text/html"`, `"text/json"`, `"image/jpg"`, `"image/gif"`, `"application/pdf"`, `"*/*"` |

**Returns:**

```
N/A
```

**Example:**

```
var file = "mSharedImage.gif"
file_copy("YYImage.jpg",file)

var read = "android.permission.READ_EXTERNAL_STORAGE"
var write = "android.permission.WRITE_EXTERNAL_STORAGE"
if(os_check_permission(write) and os_check_permission(read))
    MobileUtils_Share_Open("Title!","image/jpg",file)
else
    os_request_permission(write,read)
```

The above code will show a code example.