

dotPhone Developer Documentation

Platform Overview

dotPhone is the application interface environment for **dotOS**. This document targets the **Developer Version** and describes how applications define user interfaces, actions, and identity using structured configuration files.

UI Component Model

Applications define their interface using declarative UI components. Each component represents a visual or interactive element rendered by dotPhone.

Base Component Fields

- **name** — Unique component identifier.
- **type** — Component category: text, image, button.
- **content** — Displayed text, image reference, or label.
- **x / y** — Screen position.
- **layer** — Visual stacking order.

Text Component

Text components are used for titles, labels, and documentation-style content.

```
{  
  "type": "text",  
  "name": "title",  
  "content": "Store",  
  "textScale": "40",  
  "x": "60",  
  "y": "20"  
}
```

Image Component

Image components are commonly used for icons and branding elements.

```
{  
  "type": "image",  
  "name": "storeIcon",  
  "content": "https://example.com/store_icon.png",  
  "width": "48",  
  "height": "48",
```

```
        "x": "0",
        "y": "20",
        "layer": "1"
    }
```

Button Component

Buttons enable user interaction. Their behavior is defined by the **click** field and visual state by **texture**.

Action Convention

dotPhone uses the - symbol to define button actions.

- **-ui_name** — Navigate to another UI.
- **--code_action** — Execute application logic.
- **-system.action** — Access system-level features.

```
{
    "type": "button",
    "name": "buy",
    "content": "Buy",
    "width": "320",
    "height": "90",
    "x": "0",
    "y": "200",
    "click": "--store.buy",
    "texture": {
        "idle": "https://example.com	btn_idle.png",
        "focus": "https://example.com	btn_focus.png",
        "click": "https://example.com	btn_click.png"
    }
}
```

Example: Store UI

The example below demonstrates a simple store interface built using text, image, and button components.

```
[
{
    "type": "image",
    "name": "storeIcon",
    "content": "https://example.com/store.png",
    "width": "48",
    "height": "48",
    "x": "0",
    "y": "20"
},
```

```

{
  "type": "text",
  "name": "storeTitle",
  "content": "dotStore",
  "textScale": "42",
  "x": "60",
  "y": "20"
},
{
  "type": "text",
  "name": "item",
  "content": "Premium Theme - $2.99",
  "textScale": "24",
  "x": "0",
  "y": "100"
},
{
  "type": "button",
  "name": "buyTheme",
  "content": "Buy",
  "width": "360",
  "height": "90",
  "x": "0",
  "y": "160",
  "click": "--store.buy",
  "texture": {
    "idle": "idle.png",
    "focus": "focus.png",
    "click": "click.png"
  }
}
]

```

Application Icon and Title

Application identity is defined separately from UI files. Each project contains a configuration file that defines the app icon, title, and permission level.

```

project-name/
  conf.json
  principal-gui.json
  principal-code.json
  gui/
    code/

```

conf.json Structure

```

{
  "icon": "https://example.com/app_icon.png",
  "title": "My App",
  "permission": "level 1"
}

```

Developer Notes

Permissions range from level 1 to level 5 and define system access scope. System-related actions must use the **-system** prefix. UI and code logic remain separated to ensure clarity and maintainability.