📘How to Create a Basic Gutenberg Block Without React (Plain JavaScript Version)

📖Overview

This guide will walk you through building a custom Gutenberg block in WordPress **without using React**, using plain JavaScript and PHP. We will break down every step using a dual-column format for **PHP** and **JavaScript**, followed by a full explanation, example images, and downloadable formats.

🔧Step-by-Step Breakdown (PHP vs JS)

PHP (Server-side)	JavaScript (Client-side)
1. Hook into WordPress	1. Register the block with `wp.blocks.registerBlockType()`
2. Register the block using `register_block_type()`	2. Define block attributes (`message`, `color`, etc.)
3. Define `render_callback` for dynamic output	3. Use `edit()` to control block UI
4. Register JS file with `enqueue_block_editor_assets`	4. `save()` returns `null` to indicate dynamic block

---

🧰Full PHP Code (Backend Rendering)

```
class Colored_Message
{
    public function __construct()
    {
        add_action('init', [$this, 'init']);
        add_action('enqueue_block_editor_assets', [$this, 'load_assets']);
    }

    public function init()
    {
        register_block_type('mgb/colored-message', [
            'render_callback' => [$this, 'render_colored_message'],
            'attributes' => [
                'message' => ['type' => 'string', 'default' => 'Hello from
colorful world'],
                'color' => ['type' => 'string', 'default' => '#000'],
                'fontSize' => ['type' => 'number', 'default' => 16],
                'backgroundColor' => ['type' => 'string', 'default' => '#fff'],
            ],
        ]);
    }

    public function render_colored_message($attributes, $content)
    {
        $message = esc_html($attributes['message'] ?? 'Hello from colorful
world');
        $color = esc_attr($attributes['color'] ?? '#000');
        $fontSize = esc_attr($attributes['fontSize'] ?? 16);
        $backgroundColor = esc_attr($attributes['backgroundColor'] ?? '#fff');

        ob_start();
        ?>
        <div style="background-color:<?= $backgroundColor ?>; padding:20px;
border-radius:10px;">
            <p style="color:<?= $color ?>; font-size:<?= $fontSize ?>px; margin:
0;"> <?= $message ?> </p>
        </div>
        <?php
        return ob_get_clean();
    }

    public function load_assets()
    {
        wp_enqueue_script(
            'mgb-colored-message',
            MGB_URL . 'widget/colored-message/colored-message.js',
            ['wp-blocks', 'wp-element', 'wp-editor', 'wp-components', 'wp-
i18n'],
            '1.0.1',
            true
        );
    }
}

new Colored_Message();
```

JavaScript Code (Editor UI)

```
console.log("colored message loaded");

wp.blocks.registerBlockType("mgb/colored-message", {
  title: "Mgb Colored Message",
  icon: "dashicons-media-text",
  category: "widgets",
  attributes: {
    message: { type: "string", default: "Hello from colorful world" },
    color: { type: "string", default: "#000" },
    fontSize: { type: "number", default: 16 },
    backgroundColor: { type: "string", default: "#fff" },
  },
  edit: (props) => {
    const InspectorControls = wp.blockEditor?.InspectorControls ||
wp.editor.InspectorControls;
    const PanelBody = wp.components.PanelBody;
    const inputStyle = {
      width: "100%",
      marginBottom: "16px",
      boxSizing: "border-box",
    };

    return [
      wp.element.createElement(
        InspectorControls,
        {},
        wp.element.createElement(
          PanelBody,
          { title: "Message", initialOpen: true },
          wp.element.createElement("div", {},
            wp.element.createElement("label", {}, "Message:"),
            wp.element.createElement("input", {
              type: "text",
              value: props.attributes.message,
              onChange: (e) => props.setAttributes({ message: e.target.value }),
              style: inputStyle,
            }),
            wp.element.createElement("label", {}, "Text Color:"),
            wp.element.createElement("input", {
              type: "color",
              value: props.attributes.color,
              onChange: (e) => props.setAttributes({ color: e.target.value }),
              style: inputStyle,
            }),
            wp.element.createElement("label", {}, "Font Size (px):"),
            wp.element.createElement("input", {
              type: "number",
              value: props.attributes.fontSize,
              onChange: (e) => props.setAttributes({ fontSize:
parseInt(e.target.value, 10) }),
              style: inputStyle,
            }),
            wp.element.createElement("label", {}, "Background Color:"),
            wp.element.createElement("input", {
              type: "color",
              value: props.attributes.backgroundColor,
              onChange: (e) => props.setAttributes({ backgroundColor:
e.target.value }),
              style: inputStyle,
            })
          )
        )
      ),
      wp.element.createElement(
        "div",
        {
          style: {
            backgroundColor: props.attributes.backgroundColor,
            padding: "20px",
            borderRadius: "10px",
          },
        },
        wp.element.createElement("p", {
          style: {
            color: props.attributes.color,
            fontSize: props.attributes.fontSize + "px",
          },
        }, props.attributes.message)
      )
    ];
  },
  save: () => null, // Dynamic block rendered by PHP
});
```

📝Explanation

PHP:

- Registers the block server-side with `register_block_type()`.

- Handles dynamic rendering via `render_callback`.
- Provides default values and safe output with `esc_html()` and `esc_attr()`.
- Loads the JavaScript file using `enqueue_block_editor_assets`.

JS:

- Uses `wp.blocks.registerBlockType()` to define block.
- `edit()` handles block inspector UI using `InspectorControls`.
- No React, only `wp.element.createElement()`.
- `save()` returns `null` because the output is server-rendered (dynamic block).

📏Visual Diagram (Example Layout)

```
+-----------------------------------------------+
| Message:       [ Hello World!           ]  |
| Text Color:    [ #ff0000 ]                   |
| Font Size:     [ 18       ] px              |
| Background:    [ #fff000 ]                  |
+-----------------------------------------------+

Preview:

[ Hello World! ] in red text, 18px font, yellow background
```

📦Downloads

You can export this book as:

- 📥[Download PDF]
- 📥[Download Markdown (.md)]

Let me know if you want these files generated now.