

# How to build to a Vive Pro or Vive Pro Eye

Building a VR app for the Vive Pro Eye with Unreal Engine involves several steps, including setting up the development environment, configuring project settings, and packaging the app. Below are detailed instructions to guide you through the process:

## 1. Install Unreal Engine and Required SDKs:

- Download and install the latest version of Unreal Engine from the Epic Games website (<https://www.unrealengine.com/>).
- Ensure you have SteamVR installed on your development machine. You can download SteamVR through the Steam client (<https://store.steampowered.com/steamvr>).

## 2. Set Up SteamVR for Vive Pro Eye:

- Connect your Vive Pro Eye headset to your development machine.
- Launch SteamVR and follow the on-screen instructions to set up the headset and controllers.

## 3. Create a New VR Project:

- Launch Unreal Engine and create a new project.
- Choose the "Virtual Reality" or "VR Template" as a starting point for your Vive Pro Eye project.

## 4. Configure Project Settings:

- Go to Edit > Project Settings.
- Under Platforms > Windows, configure Windows-specific settings such as the package name and minimum supported Windows version.
- Under Platforms > Windows > VR, enable "SteamVR" as the target VR platform.
- Configure VR-specific settings like tracking method (e.g., room-scale, seated) and controller bindings.

## 5. Design VR Environment and Assets:

- Create or import 3D assets for your VR environment using external tools or Unreal Engine's built-in modeling tools.

- Set up the VR-specific player pawn (character) and camera. Make sure the camera is set to match the user's head movement.

## **6. Implement VR Interactions:**

- Set up VR-specific input bindings for the headset and controllers. Unreal Engine provides a Motion Controller component for handling controller input.
- Implement VR interactions, such as grabbing objects, interacting with buttons, and teleportation.
- For advanced interactions, you may need to use Blueprints or C++ to handle custom logic.

## **7. Optimize for Performance:**

- VR applications require high performance for a smooth experience. Optimize your assets and level design to maintain a stable frame rate (e.g., optimize textures, use LODs, reduce draw calls).
- Use level streaming techniques to manage resources efficiently, especially for large and complex environments.

## **8. Build and Run the VR App:**

- In Unreal Engine, select the "Launch" or "Package" option for Windows.
- Choose "VR Preview" as the target device and build the executable.
- Once the executable is built, it will automatically launch SteamVR, and you can put on your Vive Pro Eye headset to test the app.

## **9. Testing and Debugging:**

- Test the app thoroughly on the Vive Pro Eye headset, ensuring all VR interactions and functionalities work as intended.
- Use SteamVR's built-in debugging tools to monitor performance and resolve any issues that arise during testing.

## **10. Optimize Eye Tracking Interactions (if needed):**

- If your VR app utilizes eye tracking features of the Vive Pro Eye, ensure that the interactions and visual feedback are optimized for a seamless and comfortable user experience.
- Use Unreal Engine's Eye Tracking API to access and utilize eye tracking data in your app.

## 11. **Package the App for Distribution:**

- Once your VR app is ready for distribution, package it for Windows.
- Follow the guidelines for content distribution on the Viveport store or other distribution platforms if you plan to distribute your app to a broader audience.

Remember, Vive Pro Eye development with Unreal Engine requires attention to performance optimization and compatibility with SteamVR. Regularly test your app on the Vive Pro Eye headset to ensure a comfortable and immersive experience for users, and consider integrating eye tracking features for a more innovative user interaction.

---

Revision #2

Created 28 July 2023 09:45:41 by Darsh Kadam

Updated 28 July 2023 09:47:22 by Darsh Kadam