UI Builder

By Malachi Maldonado, Ashley Perez, Justin Satriano, Luciano Loma Lorenzana

More about UI Builder

Overview: A tool that takes in user input and generates a game overlay for RPG like games, where a user is able to pick their favorite UI from given options. We utilized genetic algorithms in order to generate multiple UIs, and generate children templates based off of parent templates that the user chooses. This process repeats until the user is satisfied with a given UI.

Theme: AI as Design Aide

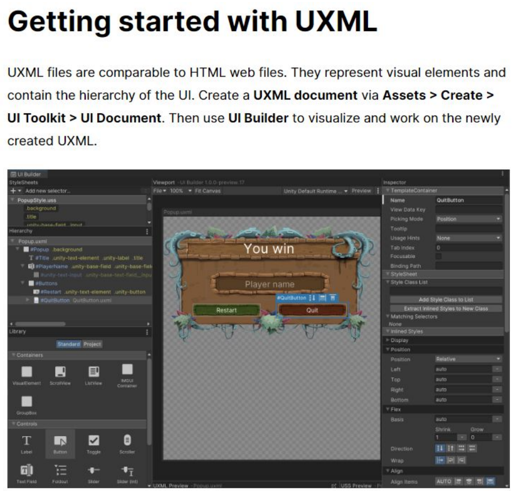
Novelty: Using AI for an overlooked but important part of game design.

Value: Create a useful AI based on the user’s input in order to help those with limited UI experience in Unity

Technology: Python, Unity/C#, Genetic Algorithms, Constraint Satisfaction

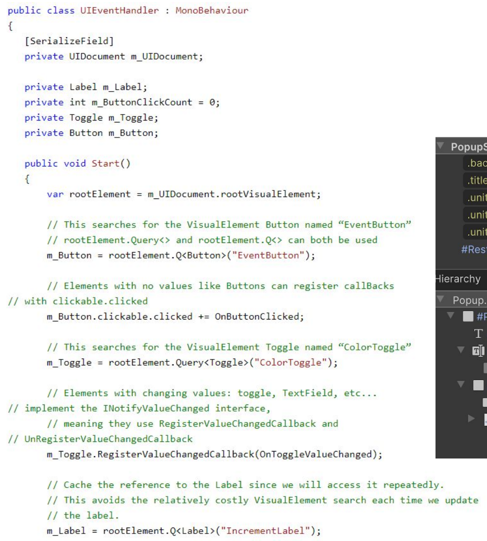
Problem Addressed

UI in video games are often overlooked and are potentially left barebones in order to focus on more “pressing” matters such as gameplay, mechanics, level design, etc. Learning about genetic algorithms we thought it would help create usable UIs that can be integrated into Unity without having to invest learning new tools that could take time away from other aspects of game development.

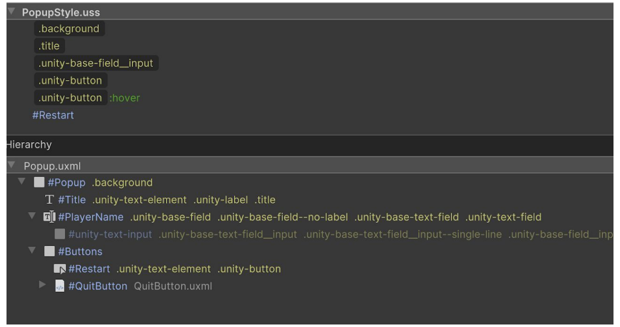


Getting started with UXML

UXML files are comparable to HTML web files. They represent visual elements and contain the hierarchy of the UI. Create a **UXML document** via **Assets > Create > UI Toolkit > UI Document**. Then use **UI Builder** to visualize and work on the newly created UXML.



```
public class UIEventHandler : MonoBehaviour
{
   [SerializeField]
   private UIDocument m_UIDocument;

   private Label m_Label;
   private int m_ButtonClickCount = 0;
   private Toggle m_Toggle;
   private Button m_Button;

   public void Start()
   {
       var rootElement = m_UIDocument.rootVisualElement;

       // This searches for the VisualElement Button named “EventButton”
       // rootElement.Query<> and rootElement.Q<> can both be used
       m_Button = rootElement.Q<Button>("EventButton");

       // Elements with no values like Buttons can register callBacks
// with clickable.clicked
       m_Button.clickable.clicked += OnButtonClicked;

       // This searches for the VisualElement Toggle named “ColorToggle”
       m_Toggle = rootElement.Query<Toggle>("ColorToggle");

       // Elements with changing values: toggle, TextField, etc...
// implement the INotifyValueChanged interface,
       // meaning they use RegisterValueChangedCallback and
// UnRegisterValueChangedCallback
       m_Toggle.RegisterValueChangedCallback(OnToggleValueChanged);

       // Cache the reference to the Label since we will access it repeatedly.
       // This avoids the relatively costly VisualElement search each time we update
       // the label.
       m_Label = rootElement.Q<Label>("IncrementLabel");
```



```
PopupStyle.uss
    .background
    .title
    .unity-base-field__input
    .unity-button
    .unity-button :hover
   #Restart
Hierarchy
Popup.uxml
  #Popup .background
    #Title .unity-text-element .unity-label .title
    #PlayerName .unity-base-field .unity-base-field--no-label .unity-base-text-field .unity-text-field
      #unity-text-input .unity-base-text-field__input .unity-base-text-field__input--single-line .unity-base-field__inp
    #Buttons
      #Restart .unity-text-element .unity-button
      #QuitButton QuitButton.uxml
```

Research

We looked into the UI field, and found some standards for what is considered a good UI.

Usability: intuitive / clear instructions and feedback

Task completion rate: the number of tasks the player is able to complete successfully and understanding what something is representing

Error rate: number of errors made using the UI / not understanding what an asset's purpose is

User satisfaction: measured through feedback like surveys

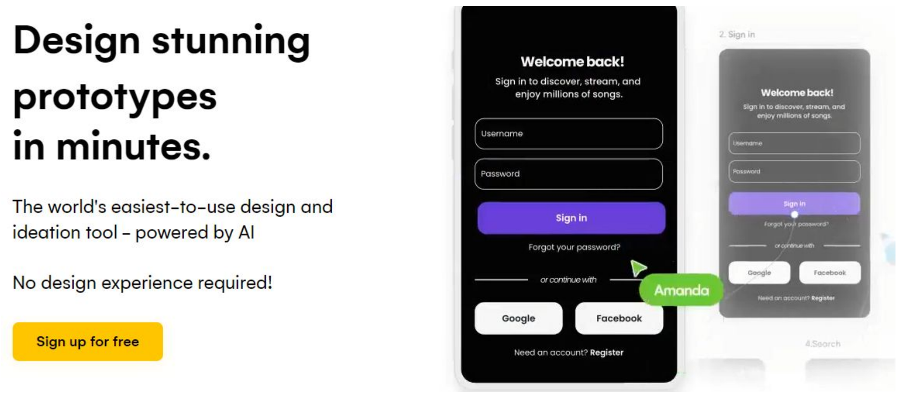
Efficiency: should be able to do desired tasks quickly as well as give the information clearly

Aesthetics: should be pleasant to look at (subjective)

Feedback: user should be able to distinguish if their actions have consequences/are successful

Novelty/Fun

Tools such as uizard exist and use AI to create UI but these are not specific to games or Unity. We wanted to create a tool that is easy to use and implements the desired functionality in order to speed up game development process.

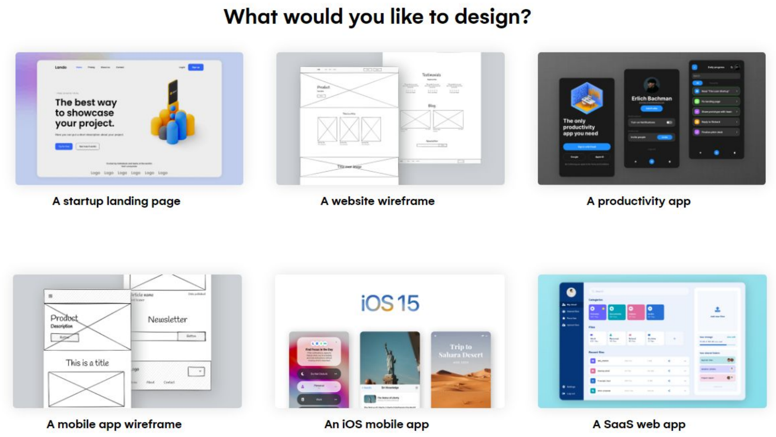


Design stunning prototypes in minutes.

The world's easiest-to-use design and ideation tool - powered by AI

No design experience required!

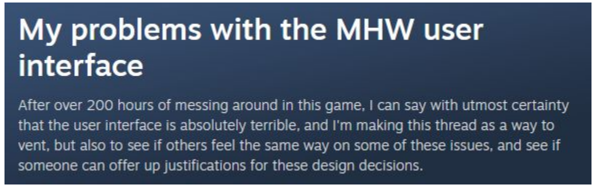
Sign up for free



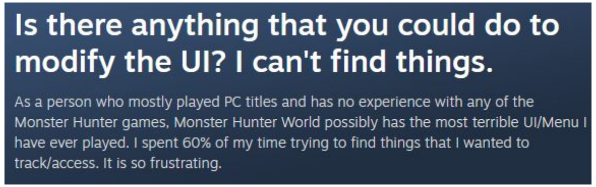
What would you like to design?

Choosing a Game

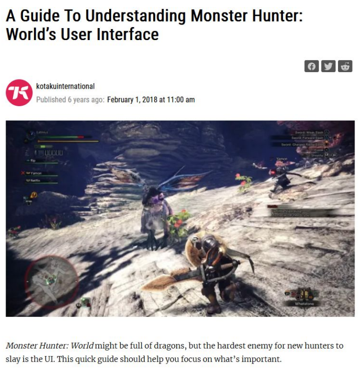
We looked for games with notable UIs, for better or worse. Some of our teammates suggested *Monster Hunter: World*, and it quickly became clear that there was potential for improvement.


My problems with the MHW user interface

After over 200 hours of messing around in this game, I can say with utmost certainty that the user interface is absolutely terrible, and I'm making this thread as a way to vent, but also to see if others feel the same way on some of these issues, and see if someone can offer up justifications for these design decisions.

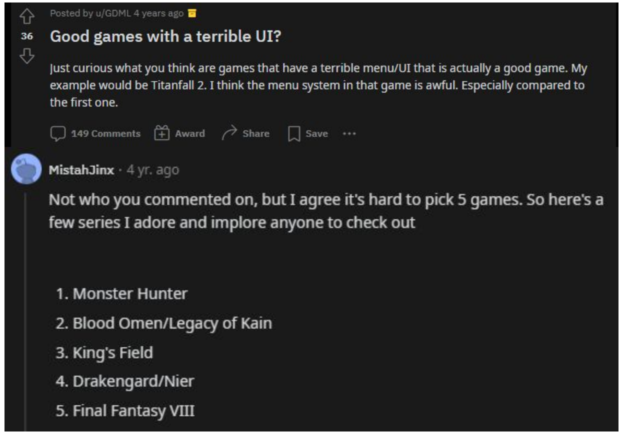

Is there anything that you could do to modify the UI? I can't find things.

As a person who mostly played PC titles and has no experience with any of the Monster Hunter games, Monster Hunter World possibly has the most terrible UI/Menu I have ever played. I spent 60% of my time trying to find things that I wanted to track/access. It is so frustrating.


A Guide To Understanding Monster Hunter: World's User Interface

kotakuinternational
Published 6 years ago: February 1, 2018 at 11:00 am

Monster Hunter: World might be full of dragons, but the hardest enemy for new hunters to slay is the UI. This quick guide should help you focus on what's important.


Posted by u/GDML 4 years ago

36

Good games with a terrible UI?

Just curious what you think are games that have a terrible menu/UI that is actually a good game. My example would be Titanfall 2. I think the menu system in that game is awful. Especially compared to the first one.

149 Comments Award Share Save ...

MistahJinx · 4 yr. ago

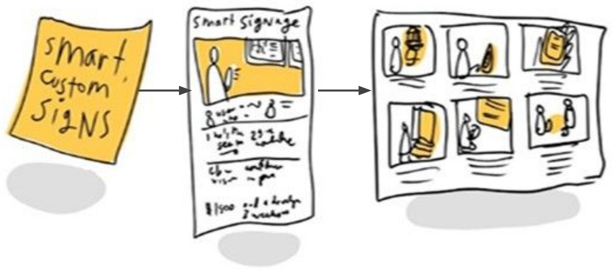
Not who you commented on, but I agree it's hard to pick 5 games. So here's a few series I adore and implore anyone to check out

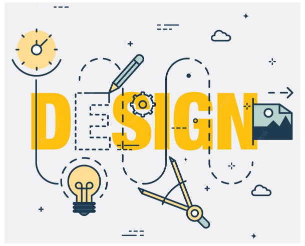
1. Monster Hunter
2. Blood Omen/Legacy of Kain
3. King's Field
4. Drakengard/Nier
5. Final Fantasy VIII

Benefits

Game designers can focus on other aspects of game development instead of trying to figure out the details of Unity's UI tools. Our tool is able to take in the user's input and create lots of variations for the user to choose from.

With our tool the user is not only able to quickly create UIs but also test them and make changes as necessary in order to listen to feedback. It can be a helpful tool not only in final deployment but also quickly making prototype UIs which is integral in creating good UI.





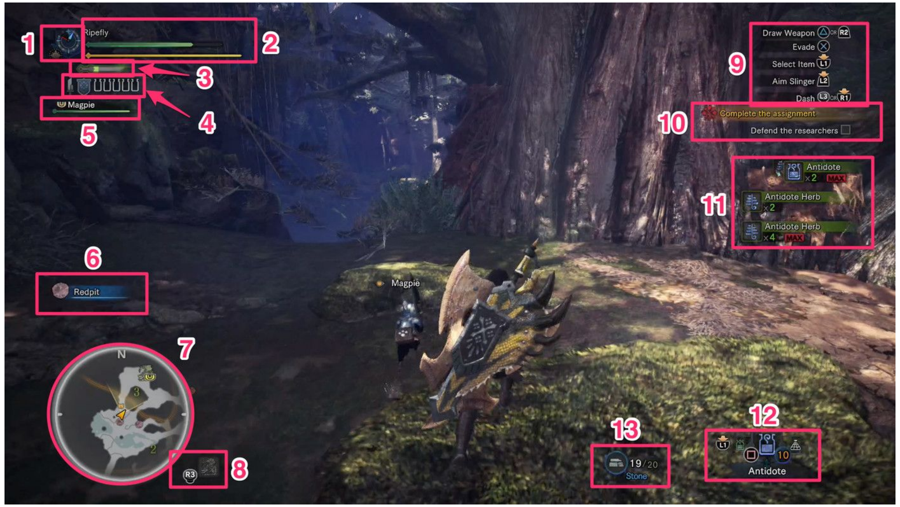
Solution

We want the user to be able to design UIs that have all the things they want but simplify the design process and ease the implementation of said UI.

The user is to specify assets they want i.e. health bar, stamina bar, mini map, etc.

From here our genetic algorithm generates UIs for the user to choose. The algorithm then uses the user's chosen UI to further generate children and keep producing UI until the user is satisfied. Then a text file is output which can be passed into Unity and implemented!

In order to create a search space to constrain our algorithm to we chose to create our tool to specifically create overlays for RPG games like Monster Hunter.



Demo