# Photon Chat UI

# Overview

Photon Chat UI provides you an easy way to add advanced chat to your Unity game. It requires the latest Photon Unity Networking package which is available to download here.

# Quick Guide

In order to use Photon Chat UI you need to drag prefab located in "*Assets/Photon Chat UI/Prefabs/Chat*" right onto your scene Canvas. Now you are ready to configure some basic settings of Chat. First of all you need to configure "Chat" component - insert your "App Id" which can be found in your Photon Chat Dashboard. Of course you can also change some settings like "App Version" or "Chat Region" which descriptions can be found in tooltips. And that's all! It should be working now. Check it out by hitting play button.

# Advanced configuration

If you want to configure or modify Photon Chat UI in more advanced way here you will find descriptions that might help you doing that. Descriptions are split into three paragraphs:

- component description
- fields description
    - required fields are written with **bold** font
    - underlined means that field require prefab object
- events description (which could be attached to Unity events like UI Button OnClick) - if event has got argument, it is described between event parenthesis. For example *eventName*(*eventArgument* as *eventArgumentType*).

For first you have to know that all of the chat UI modules are using prefabs in order to instantiate multiple copies of UI views. For example in *Chat Channel UI* you have to specify the prefab that will be used to spawn messages.

## *Chat UI*

### Description

Base component of chat. It must be added to the root of your chat. By adding *Chat UI* component to game object, component called *Chat* will be automatically added. *Chat* component configuration is described in Quick Guide chapter.

### Fields

1. **Main Dock** - reference to Main Dock component
2. **Floating Dock Prefab** - reference to floating dock prefab (more informations about floating docks can be found under *Chat Floating Dock UI* heading). Setting it to null will disable creating floating docks.
3. **Chat Channel Prefab** - reference to chat channel prefab (more informations about chat channels can be found under *Chat Channel UI* heading)
4. Channels To Subscribe At Start - list of channels that will be created/subscribed after connecting and authorizing to the chat

### Events

1. ***Connect***(*login* as ***string*** or ***InputField***) - connects to the chat server and authorizes with specified login. **Note that if you want to use password authorization you need to call Connect(ExitGames.Client.Photon.Chat.AuthenticationValues) from your code.**
2. ***CreatePublicChannel***(*channelName* as ***string*** or ***InputField)*** - creates public channel and subscribes to it. If channel already exists, it is only focused.
3. ***CreatePrivateChannel***(*username* as ***string*** or ***InputField)*** - creates private channel with another user and subscribes to it. If channel already exists, it is only focused.

## *Chat UI Auto Login*

### Description

Component used for auto logging with nick set to *PhotonNetwork.playerName*. Simply attach it to game object which contains *ChatUI* component to enable this feature.

## *Chat Channel UI*

### Description

Component used for channel display.

### Fields

1. **MessagePrefab - reference to message prefab (more informations about messages can be found under** *Chat Channel Message UI* heading)

2. **MessagesContainer** - reference to messages container - position of messages must be set automaticaly, so it's highly recommended to use one of Layout Group components to maintain it.

### Events

1. ***Close***() - closes channels.
2. ***ClearMessages***() - clears all of the messages inside channel.

## *Chat Channel Message UI*

### Description

Converts message data to readable visual form. Supports emoticons and opening private message to sender by clicking on the message.

### Fields

1. **Text** - reference to text component which will be used for displaying message. **Note that if you want to use SenderColor and MessageColor fields, Text component must have *Rich Text* property enabled.**
2. **Message Format** - format of message. **{0}** argument is replaced by message sender, **{1}** argument is replaced by message content.
3. Sender Color - color of sender text.
4. Message Color - color of message content text.
5. Open Private Chat With Sender After Click - if set to true then clicking on the message opens private chat with sender.
6. Emoticons - list of emoticons. While creating new emoticon you need to specify
    a. Sprite - the image that will be used to display emoticon
    b. Tag - the text that will be replaced by emoticon sprite
    c. Width - width of emoticon (1 unit = width of whitespace characters

## *Chat Dock UI*

### Description

Chat dock used for holding multiple channels. Only one channel can be active inside one dock.

### Fields

1. Title Text - text that will be set to title of dock. Title of dock is made of all containing channel names separated by comma - for example “Main, Global, Blue Team”.
2. **Message Input Field** - input field used to type message. Message is sent when input field is focused and Return(Enter) key is pressed.
3. **Channels Container** - place where channels will be placed after spawning. Please remember that only one channel is active in one time - because of that all of the channels can be placed in the same positions.
4. Dock Toolbar Button Prefab - reference to dock toolbar button prefab (more informations about dock toolbar buttons can be found under *Chat Dock Toolbar Button UI* heading)
5. Dock Toolbar - reference to container where toolbar buttons are spawned - position of toolbar buttons must be set automaticaly, so it’s highly recommended to use one of Layout Group components to maintain it.
6. Unread Badge - reference to game object that will be activated when channel will contain unread messages.
7. Unread Badge Amount Text - reference to Text that will be set to amount of unread messages. Note that if there are more than 9 unread messages text is set to “9+”.

### Events

1. ***Activate***(*channel* as ***ChatChannelUI***) - activates given channel.
2. ***ActivateNext***() - activates next channel.
3. ***ActivatePrevious***() - activates previous channel.
4. ***PublishMessage***(*message* as ***string*** or ***InputField***) - publishes message to active channel - in case of ***InputField*** it clears it’s content.
5. ***PublishMessage***() - publishes message from field “Message Input Field” to active channel and clears it’s content.
6. ***ClearMessages***() - clears all of the messages inside active channel.

## *Chat Floating Dock UI (based on Chat Dock UI)*

### Description

Floating dock that can be dragged or resized because it requires Chat Panel UI component.

### Events

1. **Close**() - closes the floating dock. All docked channels will be moved main dock (set in Chat UI component)

## *Chat Dock Toolbar Button UI*

### Decription

Dock toolbar button. Created in dock toolbar for every channel. Mainly used for activating and closing channels in dock. Dock toolbar buttons can be also draggable which allows user to change channel dock location.

### Fields

1. Text - reference to Text component that is used for displaying channel title. Text color is maitained by this component (by using EnabledColor and DisabledColor fields)
2. Button - reference to Button component used for activating channel. When channel is already activated then button is made not interactable.
3. Close Button - reference to Button used for closing channel. Button color is maitained by this component (by using EnabledColor and DisabledColor fields)
4. Enabled Color - color that will be set to Text and Close Button fields when channel is currently activated.
5. Disabled Color - color that will be set to Text and Close Button fields when channel isn't activated.
6. Public Channel Display Format - format of public channel title. **{0}** argument is used for channel name.
7. Private Channel Display Format - format of private channel title. **{0}** argument is used for private chat username.
8. Unread Badge - reference to game object that will be activated when channel will contain unread messages.
9. Unread Badge Amount Text - reference to Text that will be set to amount of unread messages. Note that if there are more than 9 unread messages text is set to "9+".
10. Draggable - if set to true then dock toolbar button can be dragged into another dock in order to re-dock the channel. In case when dock toolbar button isn't dropped on any dock, floating channel is created (from prefab specified in Chat UI - note that it will only work if Floating Dock Prefab isn't set to null).

### Events

1. ***Close***() - closes the channel.
2. ***Activate***() - activates the channel.

## Chat Emoticons Selector UI

### Description

Visual selector for emoticons which are spawned as buttons inside this game object. Position of emoticon buttons must be set automaticaly, so it's highly recommended to use one of Layout Group components to maintain it. By clicking on emoticon button, emoticon Tag is added to specified dock's message input field.

### Fields

1. **Dock** - dock for which emoticons selector is created.

## Chat Connection Status UI

### Description

Sets specified animator boolean parameter to true if chat is online. **If Override Animator field isn't specified then requires *Animator* component to be attached to game object.**

### Fields

1. **IsOnlineParameterName** - animator's boolean parameter which will be set to true if chat is online
2. Override Animator - if specified component will use this animator instead of the one attached to the game object.

# Chat Panel UI

## Description

UI panel that can be opened, closed, moved or resized. All of these options are optional (except opening and closing which are enabled by default).

## Fields

1. Is Resizable - if set to true then panel will be resizable
    1.1. Resize Handle Anchors - anchors which can be used for panel resizing
    1.2. Resize Handle Size - size of resize handles
    1.3. Rect Max Size - the maximum size of rect. Set 0 to disable.
    1.4. Rect Min Size - the minimum size of rect. Set 0 to disable.
2. Is Draggable - if set to true then panel will be draggable
3. Rect Constrain - the maximum rect area which should be respected by resizing and dragging operations
4. Use Animator - by using animator you are able to create fade in and fade out animations for the panel. Otherwise panel will be simply enabled or disabled.
    4.1. Is Opened Parameter Name - animator's boolean parameter which will be set to true if panel is opened. Insert empty text to disable this feature.
    4.2. Override animator - if specified panel will use this animator instead of the one attached to the panel game object.

## Events

1. ***Open***() - opens panel.
2. ***Close***() - closes panel.
3. ***Toggle***() - toggles panel. If panel is opened then it will be closed. If panel is closed then it will be opened.

# Chat Login Panel UI

## Description

Panel used for login display. Allows you to attach animator which could animate login process.

## Fields

1. Login Result Parameter Name - animator's float parameter which will be set to 1.0 if login result is successful, otherwise 0.0. Insert empty text to disable this feature.
2. Is Login Processing Parameter Name - animator's boolean parameter which is set to true if login is processing. Insert empty text to disable this feature.