

# DiscordSt

## API & Pharo Integration

**Juraj Kubelka**



UNIVERSIDAD DE CHILE



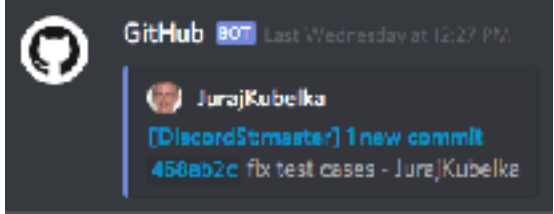
Friday, March 9, 2018

6:30 PM - 7:30 PM (UTC+02:00, Ukrainian)

5:30 PM - 6:30 PM (UTC+01:00, Paris)

01:30 PM - 02:20 PM (UTC-03:00, Chilean Time)

# TechTalk Outline

1. Motivation having **DiscordSt**.
2. **Webhook** is a low-effort way to **write messages** to channels in Discord using a secret URL. GitHub is supported directly.A screenshot of a Discord message from the GitHub Bot. The message header shows the GitHub logo, 'GitHub BOT', and the time 'Last Wednesday at 12:27 PM'. The message body shows a commit notification from 'JurajKubelka' with the text '[Discord5cmaster] 1 new commit 458ab2c fix test cases - JurajKubelka'.
3. **Bot and Client API** as a “chatbot” and standard client library to **read and write messages** and manage servers and channels. E.g., Lighthouse by Kilon: “what is pharo?”
4. **Pharo Integration.**

# TechTalk Outline

1. Motivation having **DiscordSt**.

2. **Webhook** is a low-effort way to **write messages** to channels in Discord using a secret URL. GitHub is supported directly.



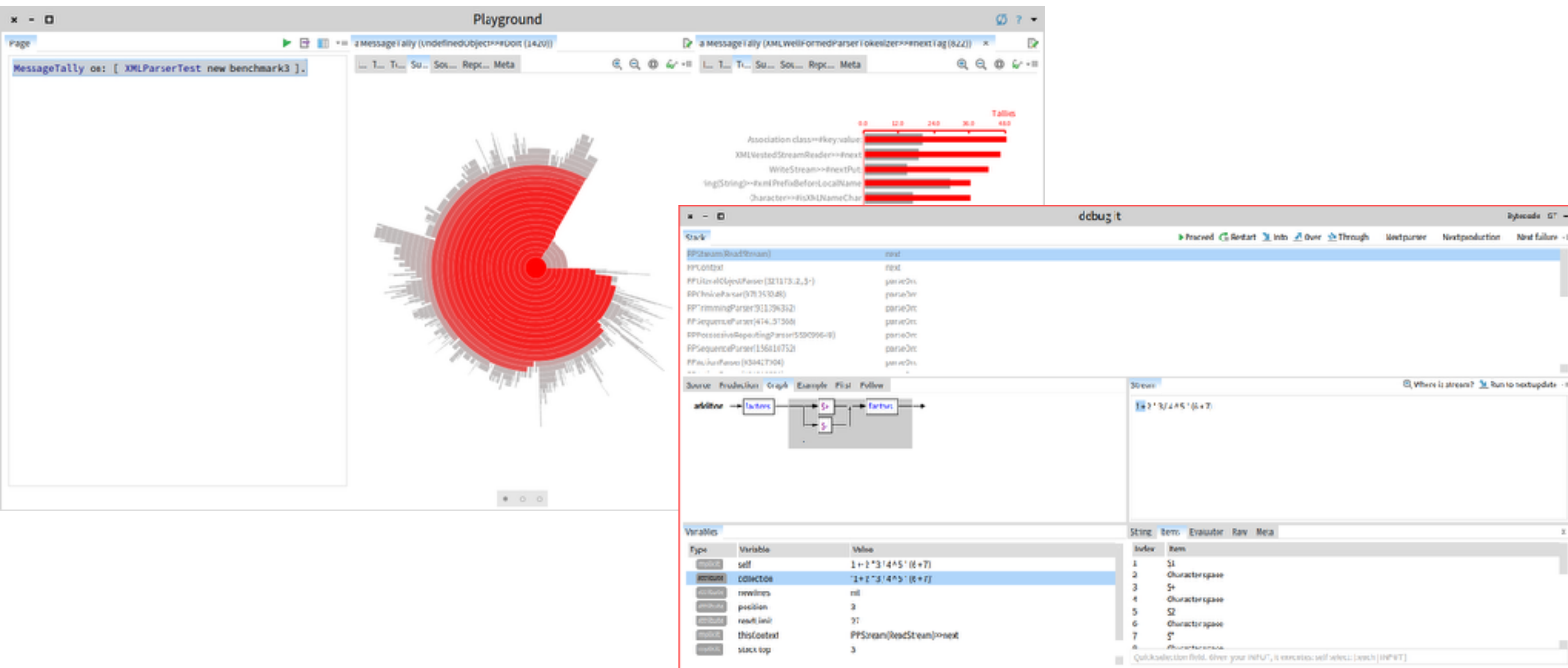
3. **Bot and Client API** as a “chatbot” and standard client library to **read and write messages** and manage servers and channels. E.g., Lighthouse by Kilon: “what is pharo?”

4. **Pharo Integration.**

# Motivation

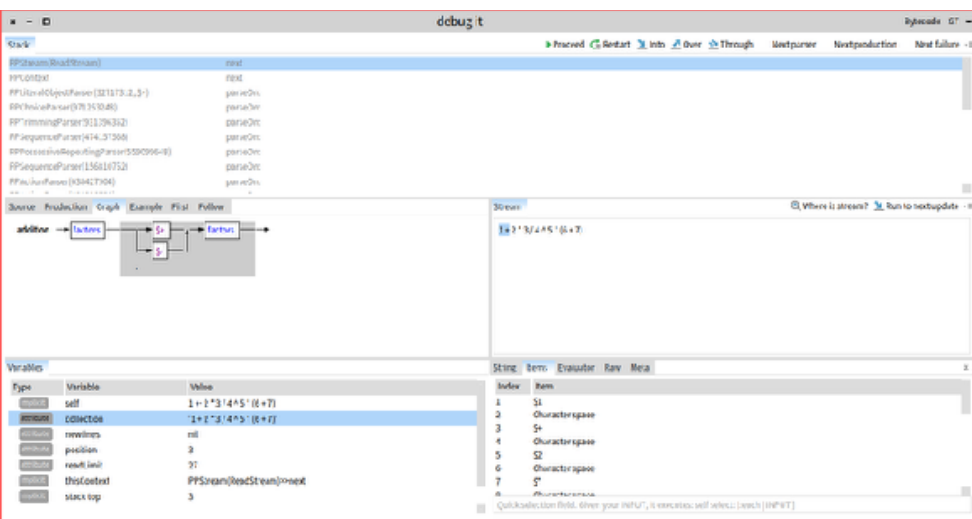
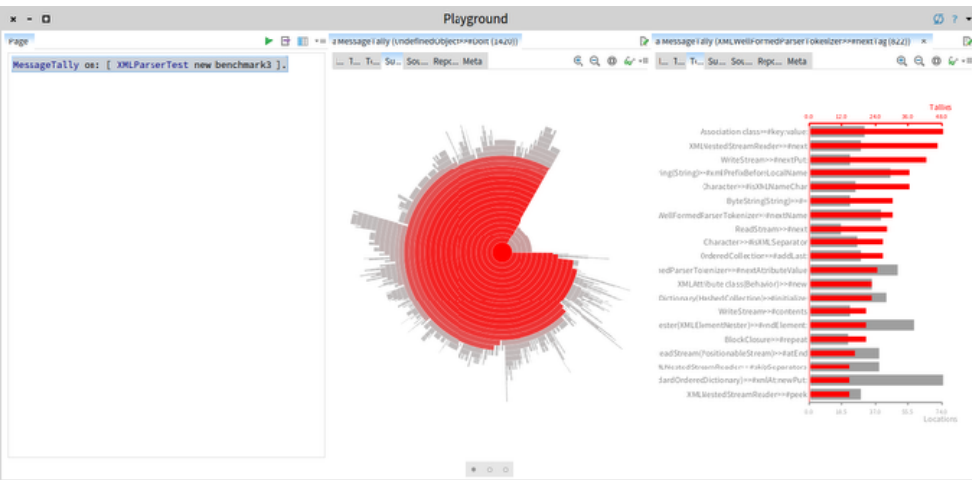
## Live Programming

frees developers from the “edit-compile-run” loop and allows people to interact with running programs very easily



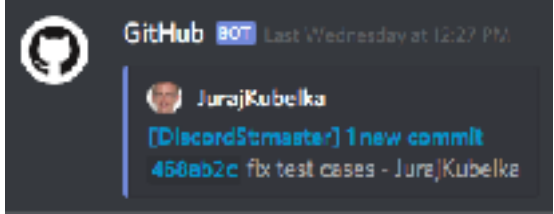
# Motivation

# How can we experience the Live Programming benefits in the developer communication?

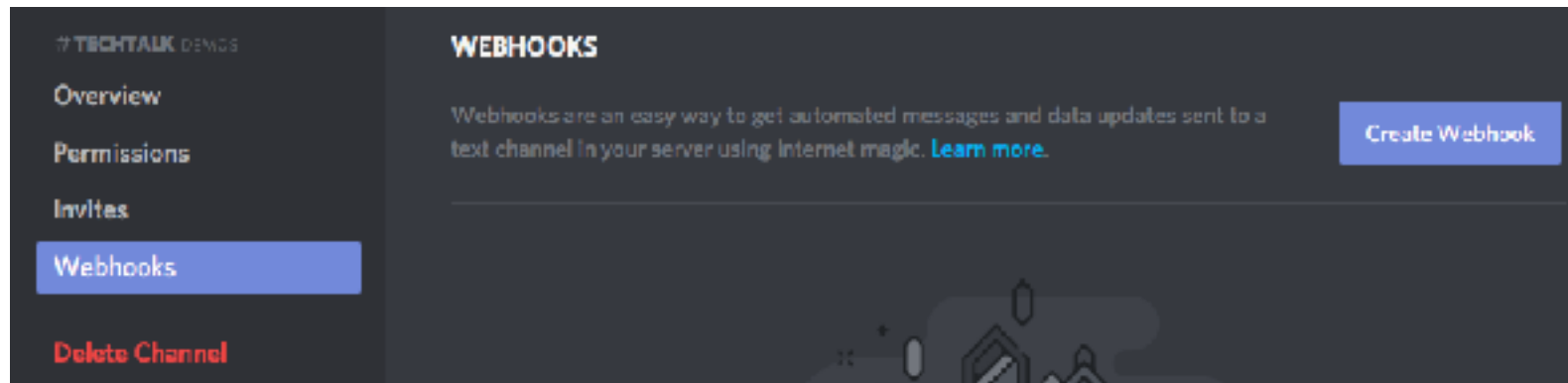
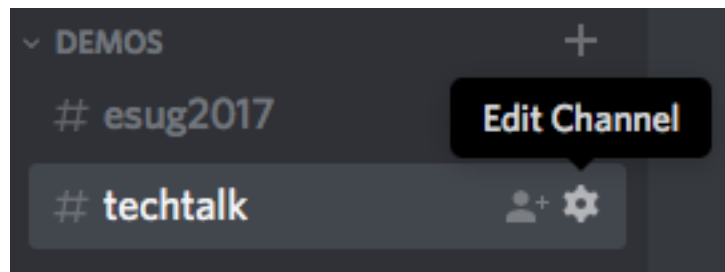


# Solution: DiscordSt

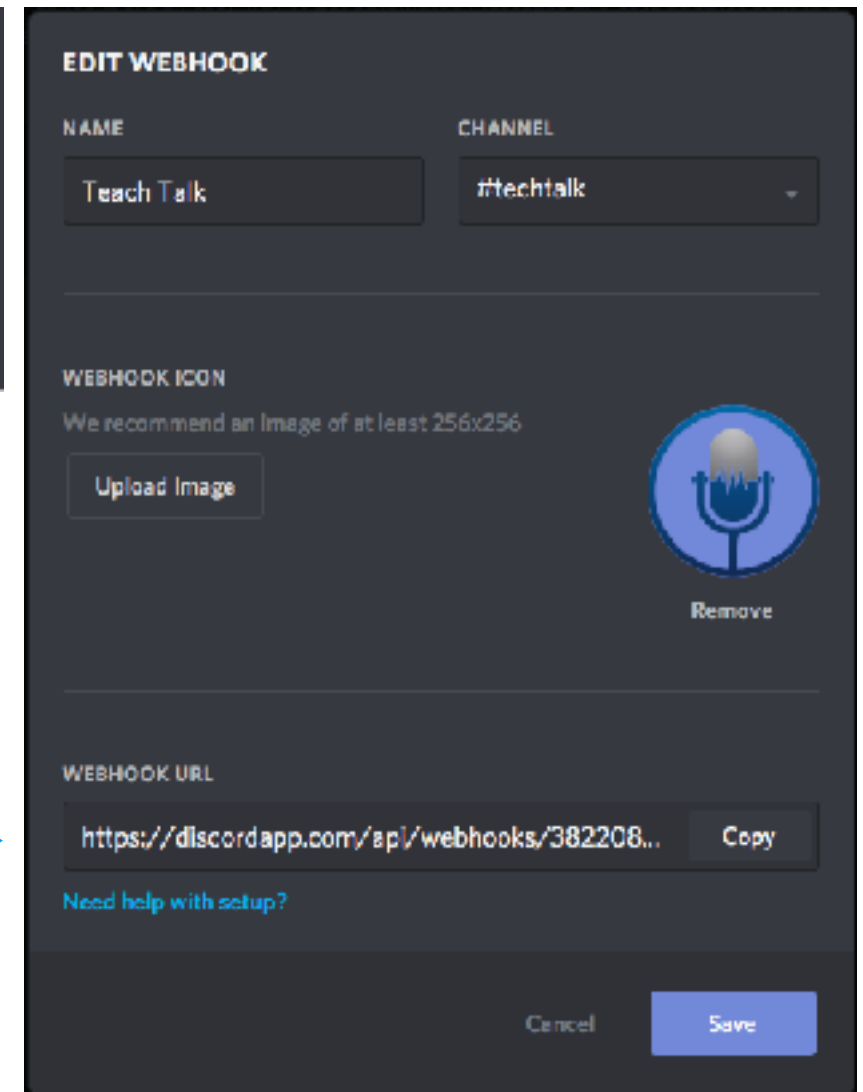
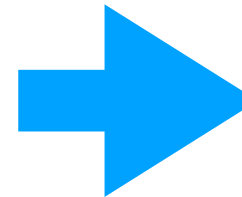
# TechTalk Outline

1. Motivation having **DiscordSt**.
2. **Webhook** is a low-effort way to **write messages** to channels in Discord using a secret URL. GitHub is supported directly.A screenshot of a Discord message from the GitHub BOT. The message is on a dark background and shows a commit notification from JurajKubelka. The text includes the GitHub logo, the bot name 'GitHub BOT', the time 'Last Wednesday at 12:27 PM', the user's name 'JurajKubelka', and the commit details '[Discord5cmaster] 1 new commit 458ab2c fix test cases - JurajKubelka'.
3. **Bot and Client API** as a “chatbot” and standard client library to **read and write messages** and manage servers and channels. E.g., Lighthouse by Kilon: “what is pharo?”
4. **Pharo Integration.**

# Webhook Configuration



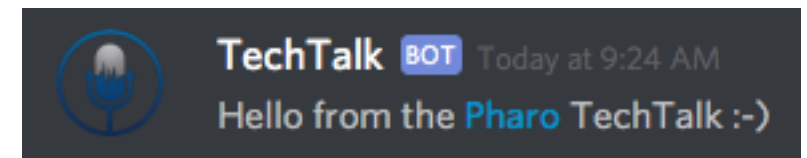
Copy the Webhook URL



# Webhook Examples

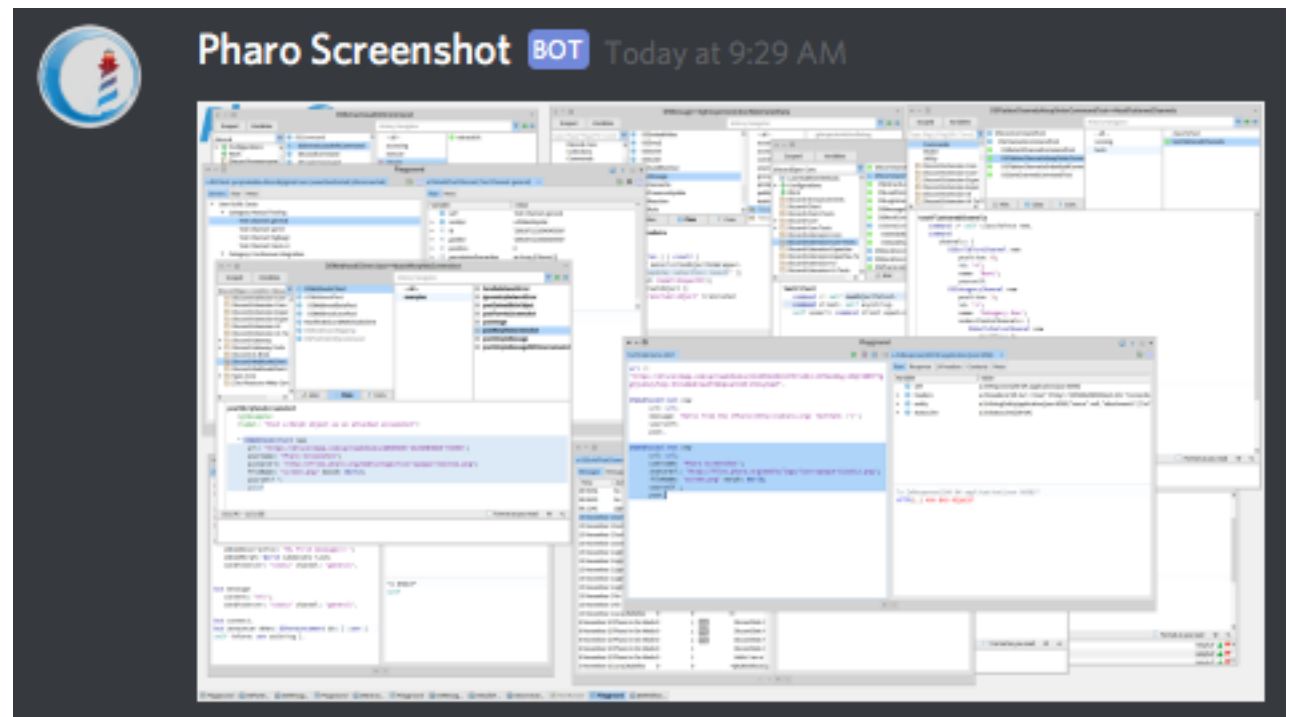
DSWebhookClient new

```
url: 'https://discordapp.com/api/webhooks/WEBHOOK-ID/WEBHOOK-TOKEN';  
message: 'Chile greets Ukraine using [Pharo](http://pharo.org) :-)';  
send.
```



DSWebhookClient new

```
url: 'https://discordapp.com/api/webhooks/WEBHOOK-ID/WEBHOOK-TOKEN';  
username: 'Pharo Screenshot';  
avatarUrl: 'http://files.pharo.org/media/logo/icon-opaque-512x512.png';  
fileName: 'screen.png' morph: World;  
send.
```





# Webhook Examples: Embed

DSWebhookClient **new**

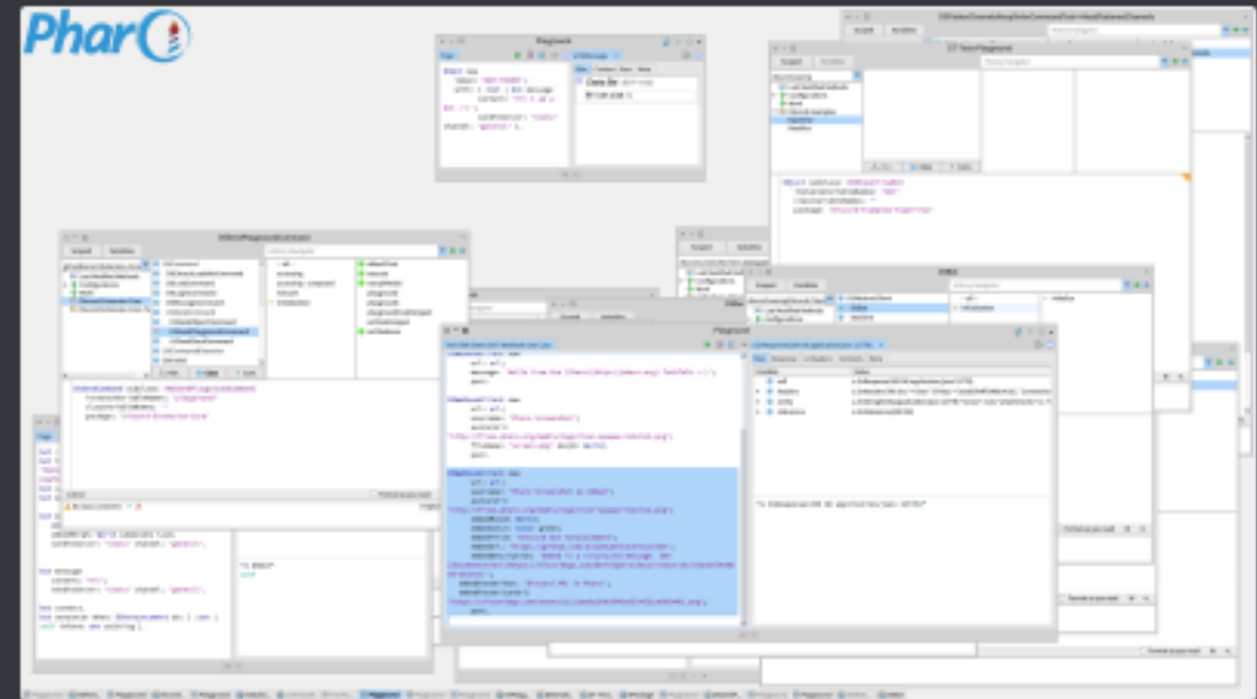
```
url: 'https://discordapp.com/api/webhooks/WEBHOOK-ID/WEBHOOK-TOKEN';  
username: 'Pharo Screenshot as Embed';  
avatarUrl: 'http://files.pharo.org/media/logo/icon-opaque-512x512.png';  
embedTitle: 'Discord Bot Announcement';  
embedDescription: 'Embed is a structured message. See [documentation](https://discordapp.com/developers/docs/resources/channel#embed-object)';  
embedUrl: 'https://github.com/JurajKubelka/DiscordSt';  
embedImageMorph: World;  
embedColor: Color green;  
embedFooterText: 'Discord API in Pharo';  
embedFooterIconUrl: 'https://discordapp.com/assets/2c21aeda16de354ba5334551a883b481.png';  
send.
```



Pharo Screenshot as Embed BOT Today at 4:02 PM

**Discord Bot Announcement**

Embed is a structured message. See [documentation](#)



Discord API in Pharo


# Roassal Script of the Day

## LIBRARIES

# databases

# polymath

# roassal

# roassal-scriptoftheday 

# seaside

# moose

# bloc

# ai

# ai-scriptoftheday

## INTERMITTENT

# techtalk

# sprint

# events

# jobs

# gci-students



## Roassal Script of the Day 10/12/2017

Script of the day: *Bundle Edges examples*

More information about Roassal on <http://agilevisualization.com/>

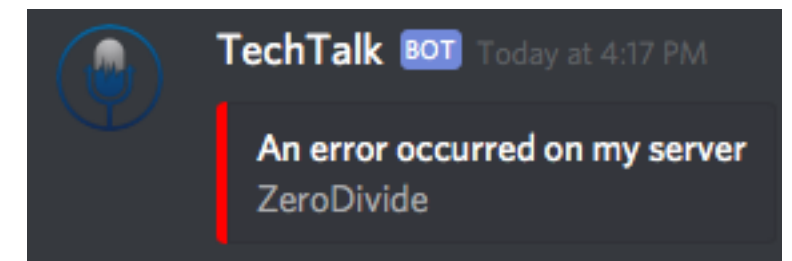
```
| b v |  
v := RTView new.  
b := RTBundleBuilder new.  
b view: v.  
b explore: Collection using: #subclasses.  
b bezier color: Color blue trans.  
b useBezierlineWith: #dependentClasses.  
b edgeBuilderDo: [ :ebuilder|  
    ebuilder shape: (RTArrowedLine new line: b bezier; yourself).  
    ebuilder connectToAll: #dependentClasses ].  
^ b
```




# Server Activity Notification



**Be informed about strange behavior on my server**



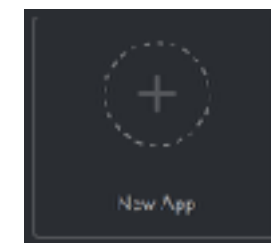
# TechTalk Outline

1. Motivation having **DiscordSt**.
2. **Webhook** is a low-effort way to **write messages** to channels in Discord using a secret URL. GitHub is supported directly.
3. **Bot and Client API** as a “chatbot” and standard client library to **read and write messages** and manage servers and channels. E.g., Lighthouse by Kilon: “what is pharo?”
4. **Pharo Integration.**


# Bot App Configuration

## Create Bot App

<https://discordapp.com/developers/applications/me>



**Bot**

A white robot head icon with green eyes.

You can bundle a Bot User with your app to interact with users in a more conversational manner. **This action is irreversible.**  
[Learn more about bot users.](#)

Create a Bot User

**APP DETAILS**  
Client ID: 382211453970808832

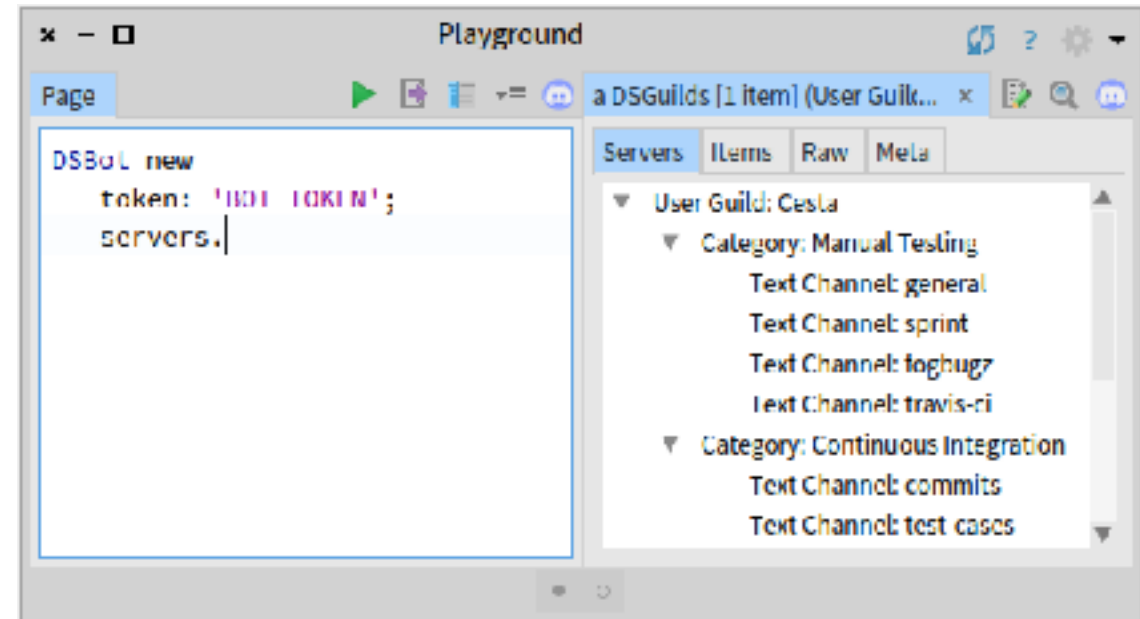
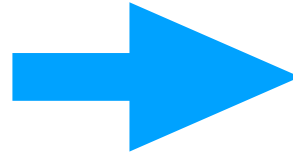
## Generate Permissions and Authorize the Bot

<https://discordapi.com/permissions.html>

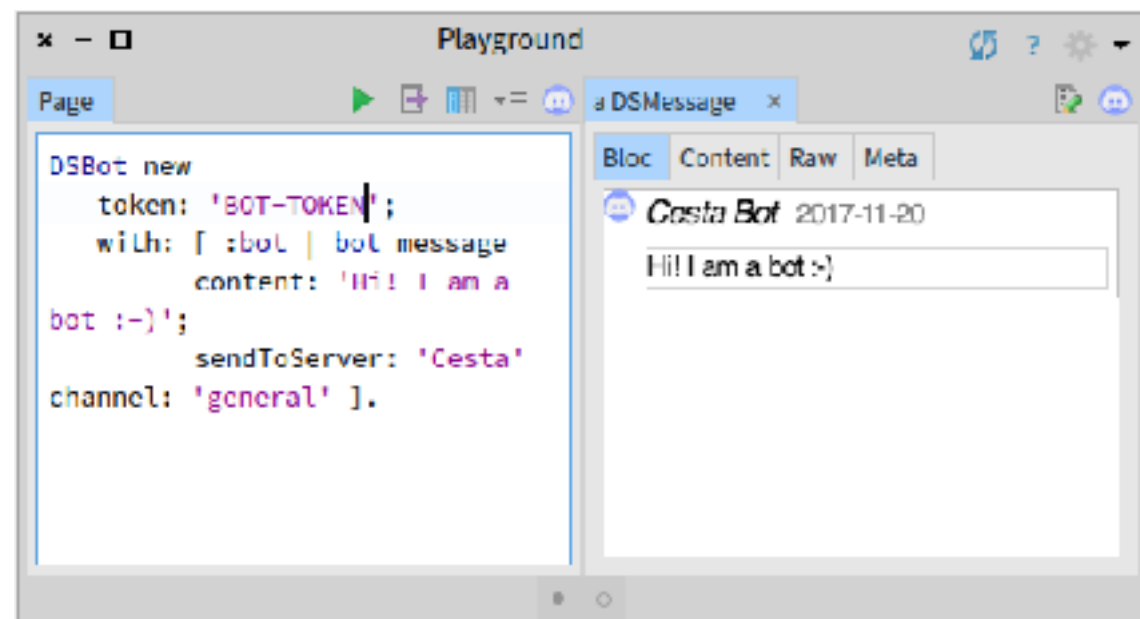
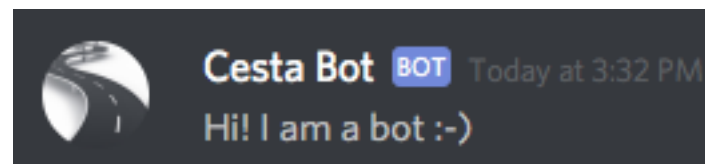
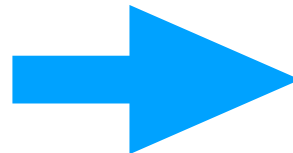
Link: [https://discordapp.com/oauth2/authorize?client\\_id=INSERT\\_CLIENT\\_ID\\_HERE&scope=bot&permissions=0](https://discordapp.com/oauth2/authorize?client_id=INSERT_CLIENT_ID_HERE&scope=bot&permissions=0)

# Bot App Examples

```
DSBot new  
token: 'BOT-TOKEN';  
servers.
```



```
DSBot new  
token: 'BOT-TOKEN';  
with: [ :bot | bot message  
content: 'Hi! I am a bot :-)';  
sendToServer: 'Pharo' channel: 'general' ].
```



# Bot Use Cases

**How can I install Bloc?**


**Is there a documentation about Spec?**


**Do we have a library to parse JSON?**



**Siri**

# Bot Use Case: Expertise

 **Juraj Kubelka** Today at 4:46 PM  
Who is expert in Playground?

 **Cesta Bot** BOT Today at 4:46 PM

You could ask the following developers:

- GT-Tests-Playground
  - StefanReichhart 50%
  - AndreiChis 49%
- GT-Playground
  - TudorGirba 33%
  - AliakseiSyrel 26%
  - AndreiChis 19%
  - StefanReichhart 15%
- ConfigurationOfGTPlaygroundCore
  - AndreiChis 90%

```
bot := DSBot new.
```

```
bot token: 'BOT-TOKEN'.
```

```
bot connect.
```

```
answer := DSExpertiseAnswer new bot: bot; yourself.
```

```
bot announcer
```

```
when: DSGatewayMessageAnnouncement
```

```
do: [ :ann |
```


```
    DSExpertiseQuestion
```

```
        message: ann message
```

```
        answer: answer ].
```



# Client Configuration



**DISCORD**

## CREATE AN ACCOUNT

EMAIL

---

USERNAME

---

PASSWORD

---

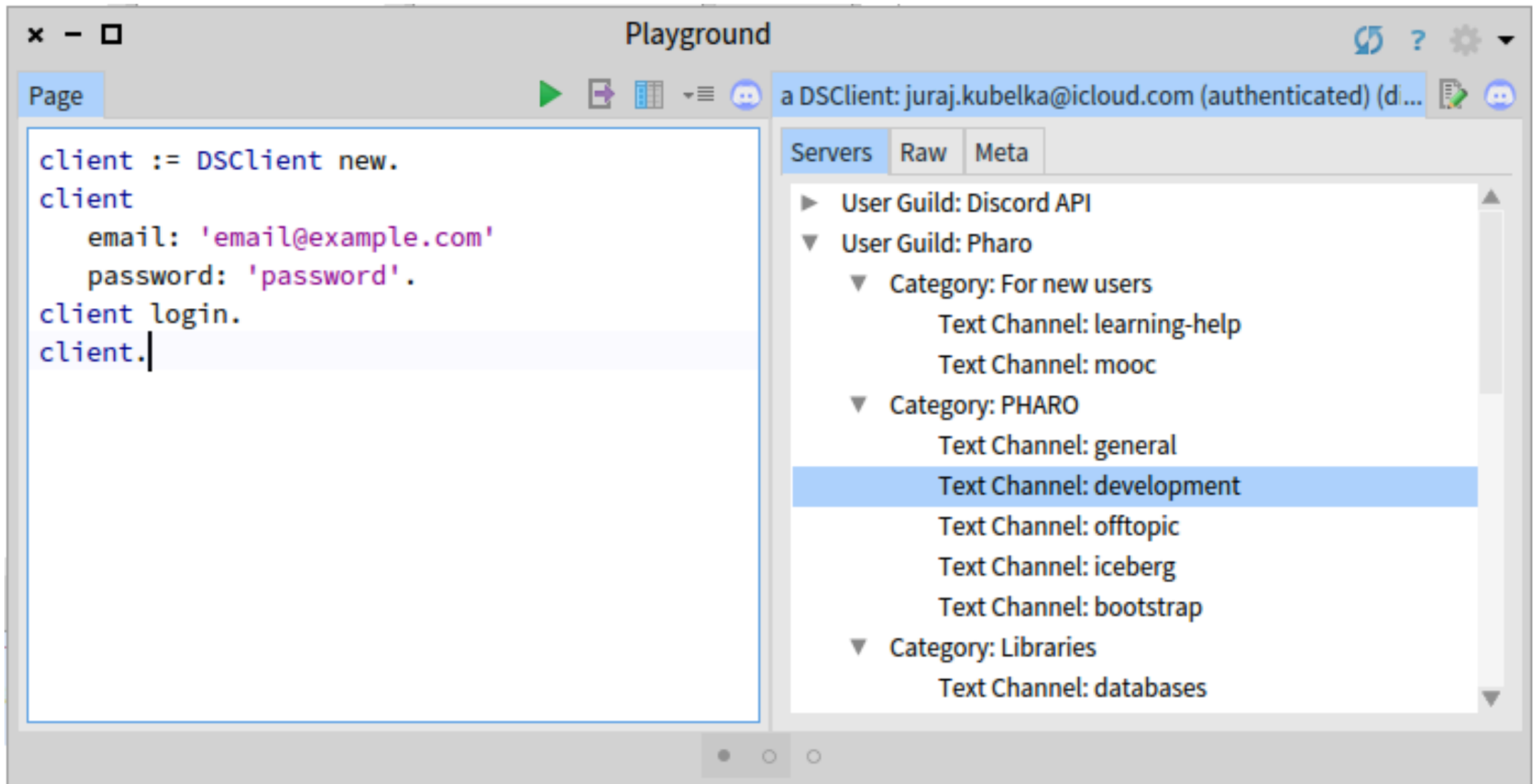
Continue

By registering, you agree to Discord's [Terms of Service](#) and [Privacy Policy](#).

---

Already have an account? [Login](#)

# Client Examples




The screenshot shows a web-based interface for testing a Discord client. The window has a title bar with standard OS controls and the word "Playground". Below the title bar is a toolbar with icons for running, saving, and other actions. The main area is split into two panes. The left pane, titled "Page", contains a code editor with the following code:

```
client := DSClient new.  
client  
  email: 'email@example.com'  
  password: 'password'.  
client login.  
client.
```

The right pane, titled "a DSClient: juraj.kubelka@icloud.com (authenticated) (di...", has tabs for "Servers", "Raw", and "Meta". The "Servers" tab is active, showing a tree view of the client's servers. The tree structure is as follows:

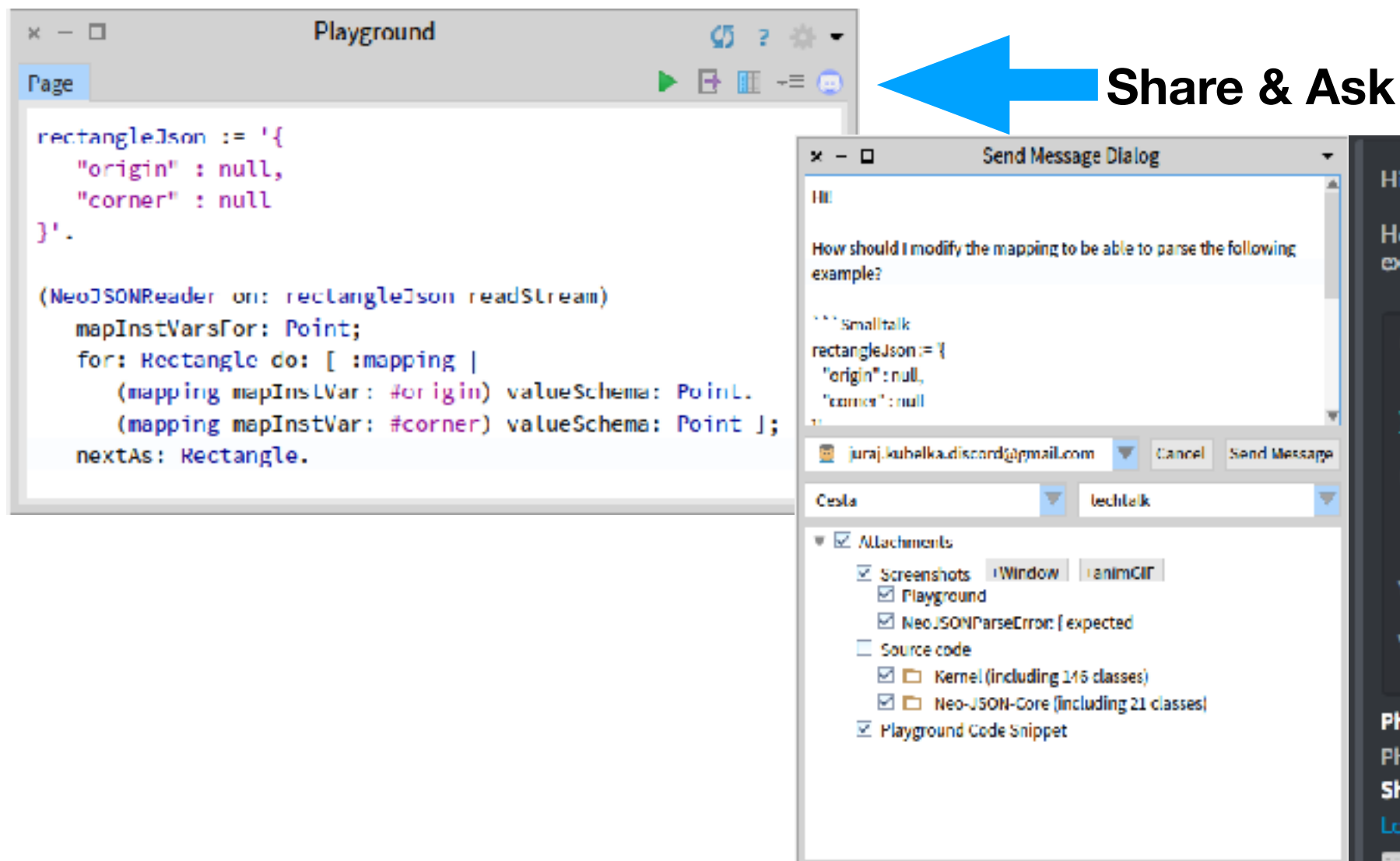
- ▶ User Guild: Discord API
- ▼ User Guild: Pharo
  - ▼ Category: For new users
    - Text Channel: learning-help
    - Text Channel: mooc
  - ▼ Category: PHARO
    - Text Channel: general
    - Text Channel: development**
    - Text Channel: offtopic
    - Text Channel: iceberg
    - Text Channel: bootstrap
  - ▼ Category: Libraries
    - Text Channel: databases

# TechTalk Outline

1. Motivation having **DiscordSt**.
2. **Webhook** is a low-effort way to **write messages** to channels in Discord using a secret URL. GitHub is supported directly.
3. **Bot and Client API** as a “chatbot” and standard client library to **read and write messages** and manage servers and channels. E.g., Lighthouse by Kilon: “what is pharo?”
4. **Pharo Integration.**

# Pharo Integration: Ask

Don't you know how to solve your task? Ask!



The image shows a Pharo Playground window on the left with the following code:

```
rectangleJson := '{
  "origin" : null,
  "corner" : null
}';

(NeoJSONReader on: rectangleJson readStream)
  mapInstVarsFor: Point;
  for: Rectangle do: [ :mapping |
    (mapping mapInstVar: #origin) valueSchema: Point;
    (mapping mapInstVar: #corner) valueSchema: Point ];
  nextAs: Rectangle.
```

A blue arrow points from the text "Share & Ask" to the "Send Message Dialog" window in the center. The dialog contains the same code snippet and has a "Send Message" button. Below the dialog, there is a list of attachments including "Screenshots", "Playground", "NeoJSONParserError: [ expected", "Source code", "Kernel (including 145 classes)", "Neo-JSON-Core (including 21 classes)", and "Playground Code Snippet".

Hi,  
How should I modify the mapping to be able to parse the following example?

```
rectangleJson := '{
  "origin" : null,
  "corner" : null
}';

(NeoJSONReader on: rectangleJson readStream)
  mapInstVarsFor: Point;
  for: Rectangle do: [ :mapping |
    (mapping mapInstVar: #origin)
      valueSchema: Point;
    (mapping mapInstVar: #corner)
      valueSchema: Point ];
  nextAs: Rectangle.
```

Pharo OS  
Pharo6.0 - 60539 (32bits) Mac OS - 1013.3

Share  
[Load in Pharo](#)



# Pharo Integration: Answer

Hi,

How should I modify the mapping to be able to parse the following example?

```
rectangleJson := '{
  "origin" : null,
  "corner" : null
}.'.

(NeoJSONReader on: rectangleJson readStream)
  mapInstVarsFor: Point;
  for: Rectangle do: [ :mapping |
    (mapping mapInstVar: #origin)
      valueSchema: Point.
    (mapping mapInstVar: #corner)
      valueSchema: Point ];
  nextAs: Rectangle.
```

Pharo

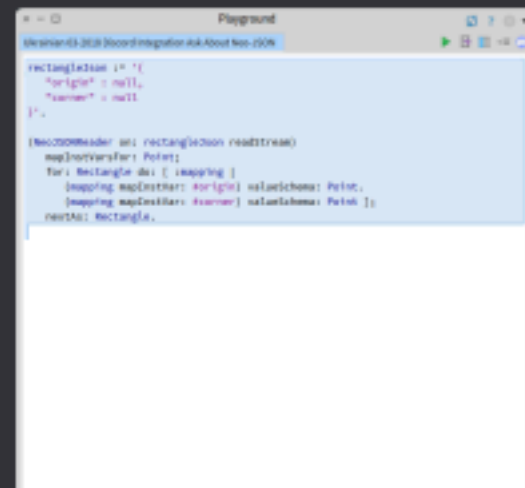
Pharo6.0 - 60539 (32bits)

OS

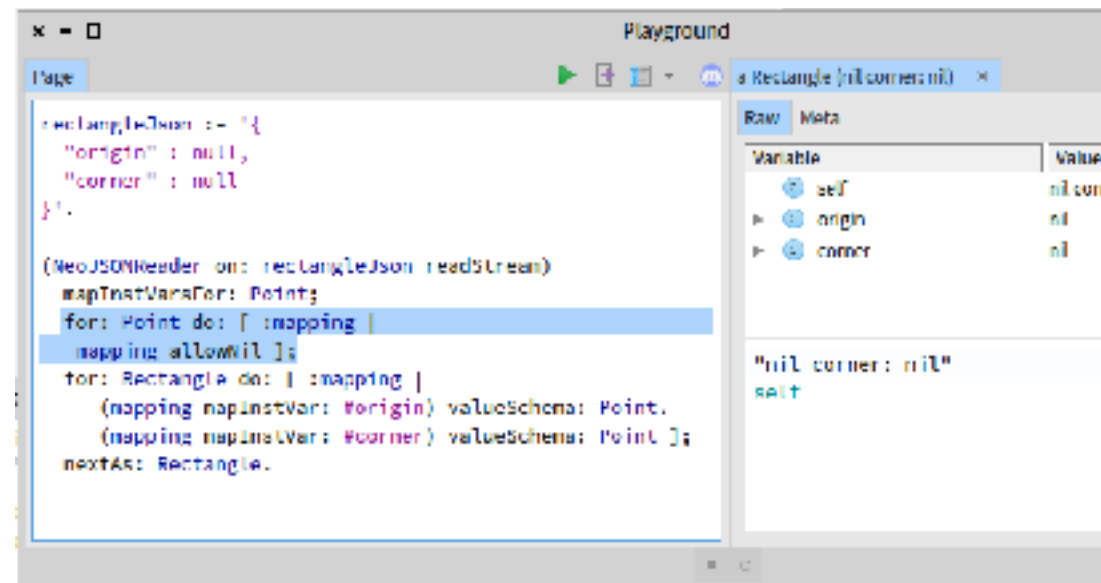
Mac OS - 1013.3

Share

[Load in Pharo](#)



## Edit & Share



```
valueSchema: Point ];
nextAs: Rectangle.
```

Pharo

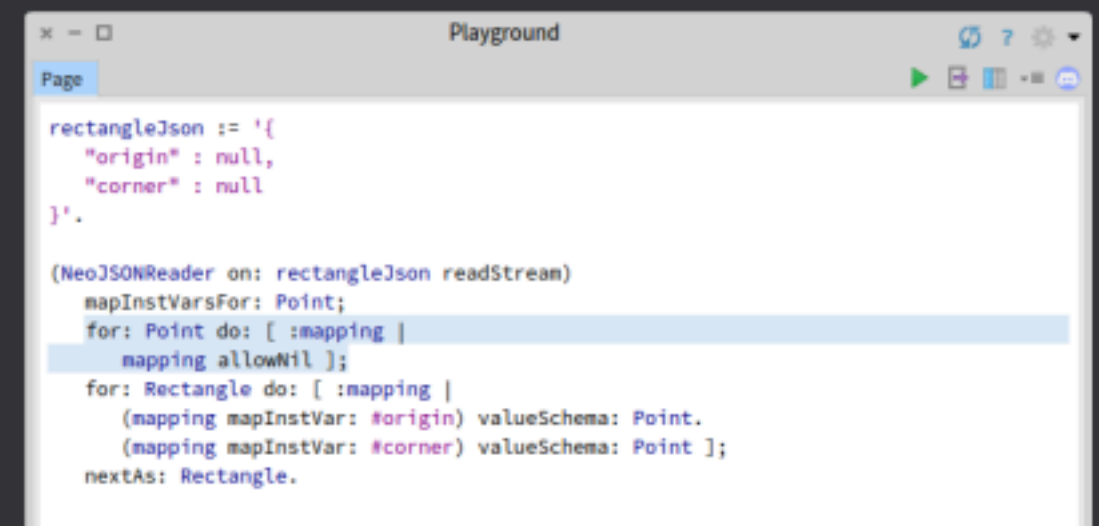
Pharo6.0 - 60539 (32bits)

OS

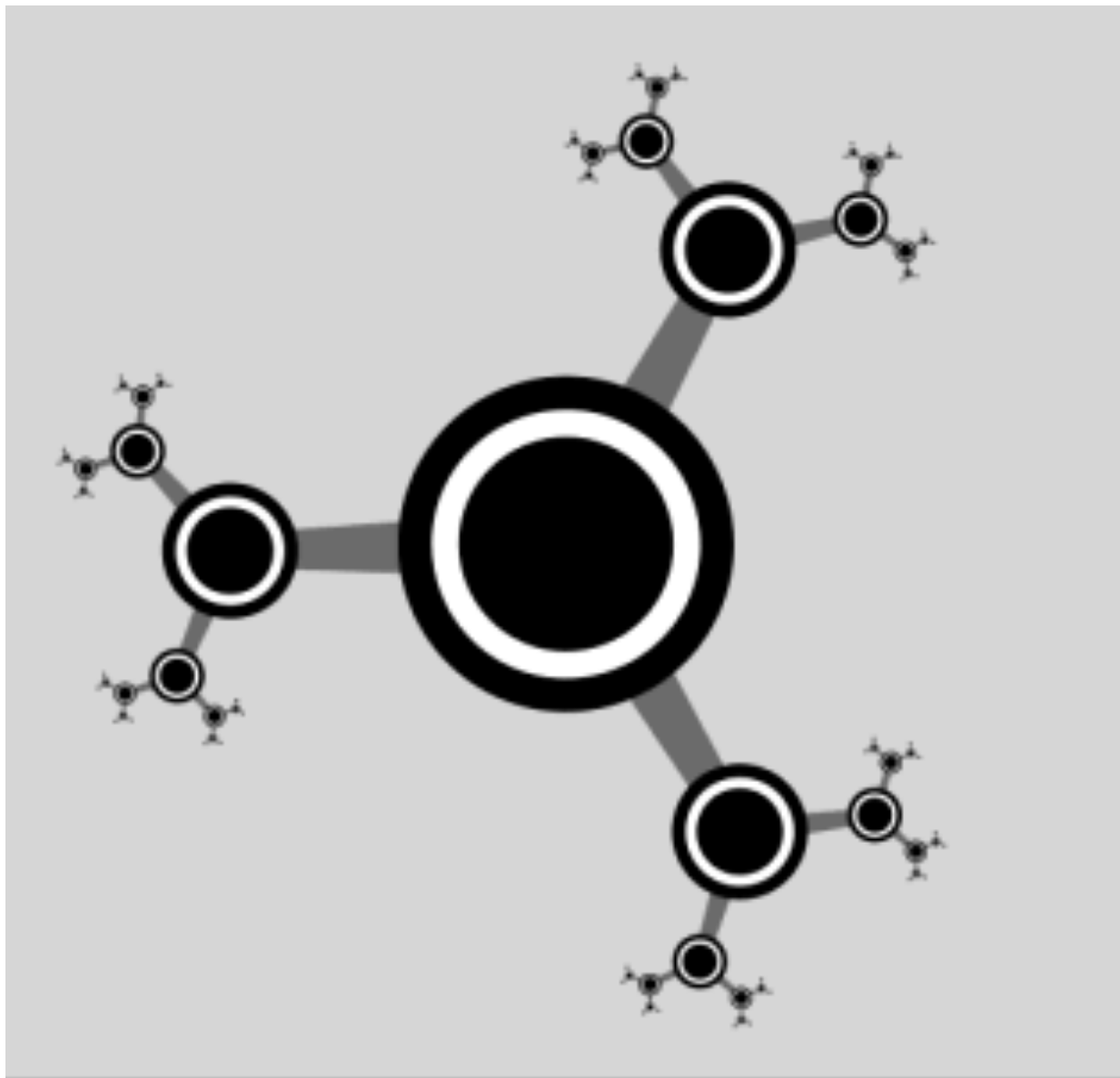
Mac OS - 1013.3

Share

[Load in Pharo](#)



# Pharo Integration: Animations



Send Message Dialog

Check this cool animation:

```
``` Smalltalk
AthensFlakeDemo new openInWindow.
```
```

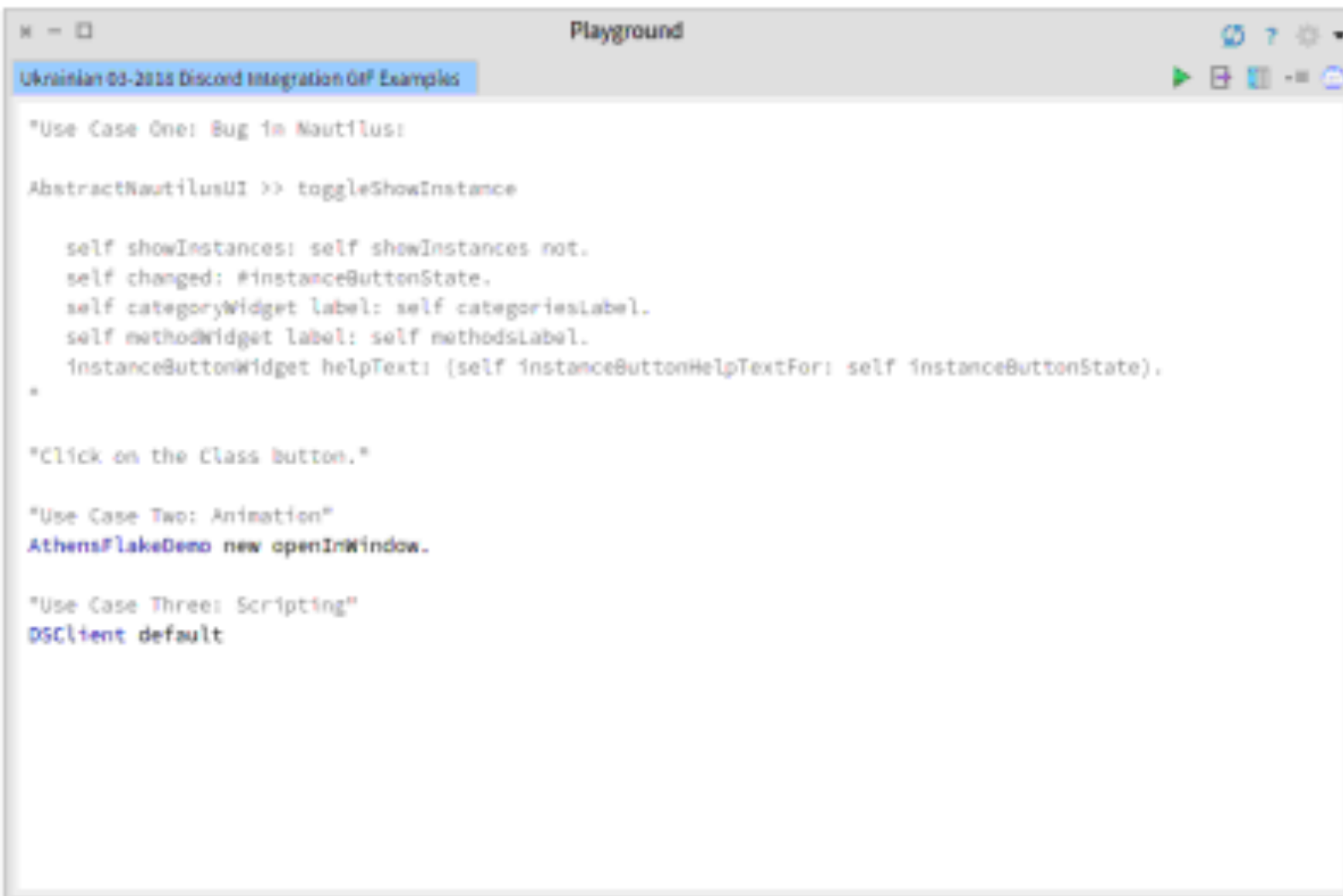
juraj.kubelka.discord@gmail.com Cancel Send Message

Cesta techtalk

Attachments

- ☒ Screenshots +Window +animGIF
- ☐ Playground
- ☒ Animated GIF Show
- ☐ Source code
- ☒ Athens-Examples (including 12 classes)
- ☒ Playground Code Snippet

# Pharo Integration: Scripting



The screenshot shows a Pharo Playground window with the title "Ukrainian 03-2018 Discord Integration Off Examples". It contains several Smalltalk code snippets:

```
"Use Case One: Bug in Nautilus:

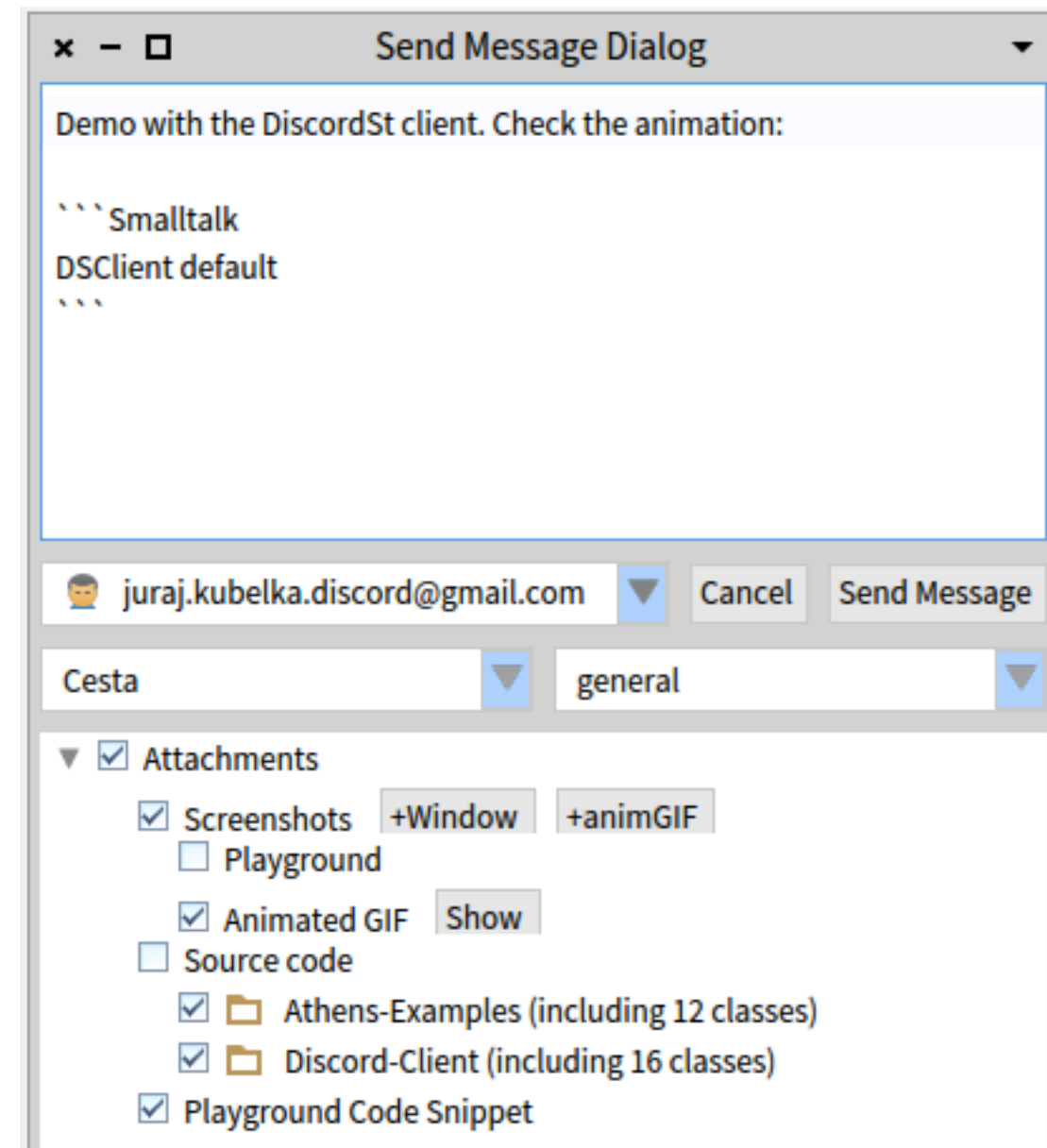
AbstractNautilusUI >> toggleShowInstance

self showInstances: self showInstances not.
self changed: #instanceButtonState.
self categoryWidget label: self categoriesLabel.
self methodWidget label: self methodsLabel.
instanceButtonWidget helpText: (self instanceButtonHelpTextFor: self instanceButtonState).
"

"Click on the Class Button."

"Use Case Two: Animation"
AthensFlakeDemo new openInWindow.

"Use Case Three: Scripting"
DSClient default
```



The screenshot shows a "Send Message Dialog" window. It contains a text area with the following content:

```
Demo with the DiscordSt client. Check the animation:

```Smalltalk
DSClient default
```
```

Below the text area, there is a field for the recipient's email address, "juraj.kubelka.discord@gmail.com", and buttons for "Cancel" and "Send Message". Below this, there are dropdown menus for "Cesta" and "general". At the bottom, there is a section for "Attachments" with a list of items:

- ☒ Attachments
  - ☒ Screenshots +Window +animGIF
    - ☐ Playground
  - ☒ Animated GIF Show
  - ☐ Source code
    - ☒ Athens-Examples (including 12 classes)
    - ☒ Discord-Client (including 16 classes)
  - ☒ Playground Code Snippet



# Pharo Integration: Errors

The screenshot displays the Pharo IDE interface. The main window shows the **AthensCairoPath** class browser. The class list on the left includes **AthensCairoPath**, which is selected. The right pane shows the class's methods, including **asAthensShapeOn:**. The bottom pane shows the class definition:

```
FFIExternalObject subclass: #AthensCairoPath
  uses: TCairoLibrary
  instanceVariableNames: ''
  classVariableNames: ''
  package: 'Athens-Cairo-Paths'
```

A red arrow points to the **AthensCairoPath** class name in the definition, highlighting an error. The status bar at the bottom left indicates "Class not referenced".

Overlaid on the bottom right is a **Send Message Dialog** window. The message text is: "Hey! Nautilus does not work. Check the animation:". The dialog includes a "Send Message" button and a list of attachments, including "Screenshots", "Animated GIF", and "Source code".

**Exception**