Jellyfish Status Tool

UML

# Diagram

JellyfishStatusUpdate

BabyJellies

Colors

Types

Jellies

SpellBox

PostToHipChat

Roles

Program

# Flow Diagram

# Form Description

## Program.cs

### Main()

Entry point for the application

Enables visual styles, Renders text, run Jellies()

## Jellies.cs

### Instance Variables

- Color gColor1

- Color gColor2

- **PostToHipChat** ptch

- **JellyfishStatusUpdate** jellyfish

- ***isResolved***rese

- ***fullUpdateType***update

- ***location***local

### Constructor

+ **Jellies()**

#### Default settings

The Jellyfish Status Tool will load with the following settings:

1. comboBox1\_Load() : fills dropdown with HipChat room information
2. comboBox1\_SelectedIndex : sets the dropdown to not have any value to selected
3. Rese : isResolved.STATUS
4. Update : fullUpdateType.NONE
5. Local : location.NONE

### Properties

* ***isResolved* Rese**
  + *get : rese*
  + *set += value*

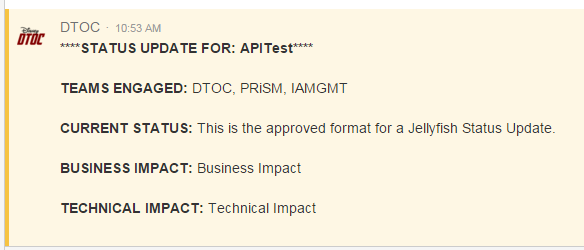
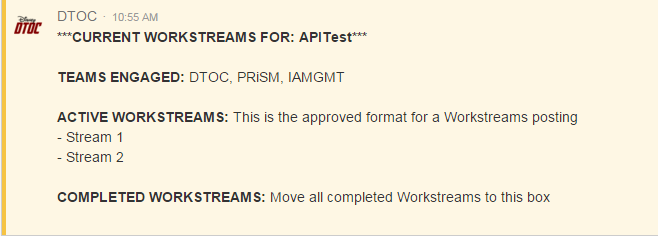
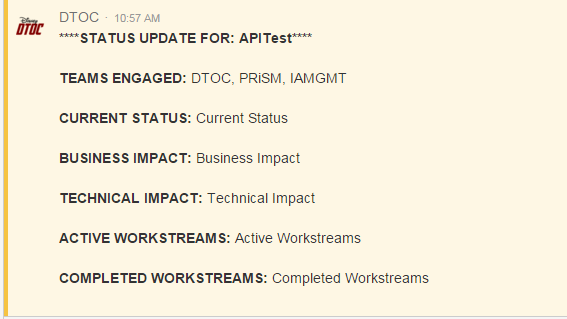
### Methods

* ***comboBox1\_Load() : void***
  + fills in the hipChat room titles
* ***comboBox1\_SelectedIndexChanged(object, EventArgs) : void***
  + sets the room id for HipChat
    - pthc.setRID(string)
  + disables comboBox1
  + sets text for textBox1
* ***button1\_Click(object, EventArgs) : void***
  + Refreshes the comboBox1 room list
    - comboBox1.Items.Clear()
    - comboBox1\_Load()
    - enables comboBox1
* ***button2\_Click(object, EventArgs) : void***
  + Clear textBox1.Text
* ***button3\_Click(object, EventArgs) : void***
  + Posting event for Incident Title
    - Sets : update.NONE
    - Sets : local.INC\_TITLE
    - cleanedJelly()
    - Sets : jellyfish
    - sendToHipChat()
  + Sends the text contained in textBox1 as a JellyfishStatusUpdate object to HipChat.
* ***button4\_Click(object, EventArgs) : void***
  + Clear textBox2.Text
* ***button5\_Click(object, EventArgs) : void***
  + Clear spellBox1.Text
* ***button6\_Click(object, EventArgs) : void***
  + Posting event for Current Status
    - Sets : update.NONE
    - Sets : local.CUR\_STATUS
    - cleanedJelly()
    - Sets : jellyfish
    - sendToHipChat()
  + Sends the text contained in spellBox1 as a JellyfishStatusUpdate object to HipChat.
* ***button7\_Click(object, EventArgs) : void***
  + Clear spellBox5.Text
* ***button8\_Click(object, EventArgs) : void***
  + Posting event for Completed Workstreams
    - Sets : update.NONE
    - Sets : local.INACTIVE\_STREAMS
    - cleanedJelly()
    - Sets : jellyfish
    - sendToHipChat()
  + Sends the text contained in spellBox5 as a JellyfishStatusUpdate() object to HipChat.
* ***button9\_Click(object, EventArgs) : void***
  + Clear spellBox2.Text
* ***button10\_Click(object, EventArgs) : void***
  + Posting event for Business Impact
    - Sets : update.NONE
    - Sets : local.BUS\_IMPACT
    - cleanedJelly()
    - Sets : jellyfish
    - sendToHipChat()
  + Sends the text contained in spellBox2 as a JellyfishStatusUpdate() object to HipChat.
* ***button11\_Click(object, EventArgs) : void***
  + Clear textBox3.Text
* ***button12\_Click(object, EventArgs) : void***
  + Posting event for The Sand Box
    - Sets : update.NONE
    - Sets : local.SAND\_BOX
    - cleanedJelly()
    - Sets : jellyfish
    - sendToHipChat()
  + Sends the text contained in textBox3 as a JellyfishStatusUpdate() object to HipChat.
* ***button13\_Click(object, EventArgs) : void***
  + Clear spellBox3.Text
* ***button14\_Click(object, EventArgs) : void***
  + Posting event for Technical Impact
    - Sets : update.NONE
    - Sets : local.TECH\_IMPACT
    - cleanedJelly()
    - Sets : jellyfish
    - sendToHipChat()
  + Sends the text contained in spellBox3 as a JellyfishStatusUpdate() object to HipChat.
* ***button15\_Click(object, EventArgs) : void***
  + Opens BabyJellies form
* ***button16\_Click(object, EventArgs) : void***
  + Posting event for Jellyfish Status
    - Sets : update.STATUS
    - Sets : local.NONE
    - cleanedJelly()
    - Sets : jellyfish
    - sendToHipChat()
  + Sends a Jellyfish Status[Title, Teams Engaged, Current Status, Business Impact, Technical Impact] as a JellyfishStatusUpdate() object to HipChat.
* ***button17\_Click(object, EventArgs) : void***
  + Clear spellBox4.Text
* ***button18\_Click(object, EventArgs) : void***
  + Posting event for Active Workstreams
    - Sets : update.NONE
    - Sets : local.ACTIVE\_STREAMS
    - cleanedJelly()
    - Sets : jellyfish
    - sendToHipChat()
  + Sends the text contained in spellBox4 as a JellyfishStatusUpdate() object to HipChat.
* ***button19\_Click(object, EventArgs) : void***
  + Posting event for Jellyfish Workstreams
    - Sets : update.WORKSTREAMS
    - Sets : local.NONE
    - cleandJelly()
    - Sets : jellyfish
    - sentToHipChat()
  + Send a Jellyfish Status[Title, Teams Engaged, Active Workstreams, Completed Workstreams] as a JellyfishStatusUpdate() object to HipChat
* ***button20\_Click(object, EventArgs) : void***
  + Posting event for Jellyfish Status Update
    - Sets : update.BOTH
    - Sets : local.NONE
    - cleanedJelly()
    - Sets : jellyfish
    - sendToHipChat()
  + Sends a Jellyfish Status [Title, Teams Engaged, Current Status, Business Impact, Technical Impact, Active Workstreams, Completed Workstreams] as a JellyfishStatusUpdate() object to HipChat.
* ***checkBox1\_isChecked(object, EventArgs) : void***
  + Controls the final and Resolved checkboxes
    - Sets : Rese.FINAL
    - Unchecks : checkBox2
* ***checkBox2\_isChecked(object, EventArgs) : void***
  + Controls the final and Resolved checkboxes
    - Sets : Rese.RESOLVED
    - Unchecks : checkBox1
* ***pictureBox1\_Click(object, EventArgs) : void***
  + Opens the Colors Form
* ***setColors(Color1, Color2) : void***
  + Called in the Colors Form
    - Sets : gColor1 = Color1
    - Sets : gColor2 = Color2
    - Adds : new PaintEventHandler(PaintEventArgs) : void
    - Invalidate : removes the previous background from the form redraws with Form1\_Paint(object, PaintEventArgs)
  + Sets the background of the Jellyfish Status Update Form
* ***Form1\_Paint(object, PaintEventArgs) : void***
  + Draws the Jellyfish Status Update form using the color choices passed to it from the Colors Form
  + There is an issue with the redraw after an analyst minimizes the form. The event is handled by the try catch. It basically forces the form to reset the paint event to the original form size.
* ***sendToHipChat() : void throws WebException***
  + Confirms a room is selected before sending to HipChat
  + Uses the url in the PostToHipChat.cs class to send formatted JellyfishStatusUpdate() object to HipChat
  + WebException will set the System.Clipboard with the text version of the JellyfishStatusUpdate() object and provide a message. \*See Troubleshooting
* ***cleanedJelly() : void*** 
  + Uses theScrubber and theCleaner to remove all hidden and special characters to prepare the JellyfishStatusUpdate() object for formatting and sending through the sendToHipChat method.
    - Sets : title = theCleaner(theScrubber(textBox1.Text))
    - Sets : engaged = theCleaner(theScrubber(textBox2.Text))
    - Sets : status = theCleaner(theScrubber(spellBox1.Text))
    - Sets : bImpact = theCleaner(theScrubber(spellBox2.Text))
    - Sets : tImpact = theCleaner(theScrubber(spellBox3.Text))
    - Sets : aWork = theCleaner(theScrubber(spellBox4.Text))
    - Sets : cWork = theCleaner(theScrubber(spellBox5.Text))
    - Sets : jellyfish = JellyfishStatusUpdate(title, engaged, status, bImpact, tImpact, aWork, cWork)
* ***theCleaner(string) : string***
  + returns empty string if parameter string is empty
    - replaces : Microsoft Smart quotes with plain text quotes
    - replaces : quotes with C# escape quotes (\\\”)
    - replaces : tab with single space
      * checks for Unicode and \t
    - replaces : Microsoft carriage return \r\n with html linebreak <br>
      * checks \r\n, \r and \n
    - replaces : Unicode backslash (\u005C) with escaped backslash (\\)
    - replaces : string \\ with \\\\ (allows single backslash character)
    - replaces : double space with single space
* ***theScrubber(string) : string***
  + Uses Regular Expressions to remove any invalid JSON character
    - Valid : !@#$%^&\*( ) \_+=- : ; “” ‘’ \ / | < , > . ? [ ] ~ 0-9 a-z A-Z

### Purpose

The Jellies Form is the main input point for users during incidents and monitoring events. It accepts the input from the users and passes that information to the helper classes. It is the parent form for all the forms in the suite.

#### Step-by-Step:

1. Open Jellyfish
2. Select the HipChat room you wish to post to
   1. Open dropdown menu located at the top of the form. (comboBox1)
   2. Choose the name of the room you are covering.
      1. If the room name is unavailable
         1. Close the drop down and press the “Get Rooms” button. (button1)
         2. This will cause the form to reload the room names from HipChat and fill the dropdown
   3. When you choose the room, the helper class (PostToHipChat.cs) will store the room id and allow you to post to only that room.
   4. The “Short Description” text box (textBox1) will fill in with the room name to act as a starting point for users.
3. Complete the “Short Description” (textBox1)
   1. This is also known as the incident title. You can leave it as is or choose to extend the information to make a more complete picture of the issue.
4. Click the “Roles” (button15)
   1. This will open the child form BabyJellies.cs
      1. The HipChat room will have to be selected before the BabyJellies.cs form will open
   2. Use the dropdown menu’s on the form to select the Analysts currently assigned to the roles.
      1. Coordinator (BabyJellies.comboBox1)
      2. Escalator (BabyJellies.comboBox2)
      3. Communicator (BabyJellies.comboBox3)
      4. DTOC SAM (BabyJellies.comboBox4)
   3. Click the “Post Roles” (BabyJellies.button1)
      1. The roles will be sent to the HipChat room you selected in the Jellyfish form.
5. Determine the teams engaged and enter them into the “Teams Engaged” (textBox2)
6. (Optional) Choose the color scheme for your Jellyfish
   1. Click on the Jellyfish Icon located next to the “Resolved” (checkBox2)
   2. This will open the child form: Colors.cs
      1. Choose your color scheme from the 10 buttons:
         1. Desert Sunset (Colors.button1)
         2. Ocean Deep (Colors.button2)
         3. Black Tie (Colors.button3)
         4. Shoreline (Colors.button4)
         5. Saturn (Colors.button5)
         6. Jellyfish Classic (Colors.button6)
         7. Dark Forest (Colors.button7)
         8. \*~Fabulous~\* (Colors.button8)
         9. Spring Time (Colors.button9)
         10. The Demon (Colors.button10)
      2. After you make your selection the Jellyfish form background will change allowing you to preview the selection.
      3. Once you find the color scheme you enjoy close the Colors.cs form
   3. The form can be opened at any time.
7. Enter the known and confirmed status into the “Current Status” box (spellBox1)
   1. The current status is a summary of the issue, what has been tried and what is currently being tried.
8. Enter the known and confirmed business impact into the “Business Impact” box (spellBox2)
   1. This is a statement of what is impaired in the business
      1. What can Cast Members not do
      2. What can Guests not do
9. Enter the known and confirmed technical impact into the “Technical Impact” box (spellBox3)
   1. Technical Impact is a statement of any known impairment of: servers, access points, routers, workstations or anything else that is affected by the outage
10. List the active workstreams in the “Active Workstreams” box (spellBox4)
    1. This sections if for the work being performed and work that is pending or being performed in parallel.
11. After a workstream has been completed transfer it to the “Completed Workstreams” box (spellBox5)
    1. This is to keep track as to where teams have been. Use your best judgment as to how long to keep past streams in this section.
12. Post to HipChat
    1. All the locations except for “Teams Engaged” can be individually posted into HipChat
    2. You can use the “Final” and “Resolved” check boxes (checkBox1 and checkBox2, respectively) to add special tags to the Jellyfish Update
       1. Final: If impact is mitigated or lowered but not completely resolved
       2. Resolved: The issue is completely fixed and teams are dropping
    3. “Post Status” (button16) This will post the “Short Description”(textBox1), “Teams Engaged”(textBox2), “Current Status”(spellBox1), “Business Impact”(spellBox2) and “Technical Impact”(spellBox3) to HipChat in the approved format:
    4. “Post Workstreams”(button19) This will post the “Short Description”(textBox1), “Teams Engaged”(textBox2), “Active Workstreams”(spellBox4), “Completed Workstreams”(spellBox5) to HipChat in the approved format:
    5. “Post All”(button20) This will post the “Status”(button16) and “Workstreams”(button19) to HipChat in the approved format:
    6. Continue to maintain and update all the fields as the Incident continues and follow the standard guidelines of posting Jellyfish updates:
       1. Priority 1: 30 min “Status” after initial, 15 min “Workstreams” after initial
       2. Priority 2: 1 hour “Status” after initial, 30 min “Workstreams” after initial
          1. Use your best judgment while you are on the bridge to determine if the posting needs to be more or less frequent.
13. “The Sand Box” (textBox3) can be used to store important information such as server names. The “Post” (button12) will send all the information contained in the box to HipChat.
    1. Try to only use this for actual important information if you plan to post it to HipChat

## BabyJellies.cs

### Instance Variable

* Roles roles
* PostToHipChat post
* string roomName

### Constructor

+ ***BabyJellies(string)***

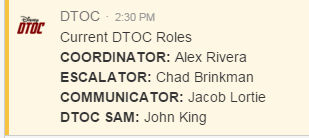
#### Default Settings

When the BabyJellies form is called the HipChat room name is used as an argument in the constructor. The room name is then passed into the PostToHipChat class via the setRID(string) : void method.

### Methods

* ***button1\_Click(object, EventArgs) : void throws WebException***
  + Gathers all the comboBox1-4 information and creates a Roles object
    - checkRoles() : bool
      * Confirms the dropdown menus are assigned
    - Sets : roles(Coordinator[comboBox1.Text], Escalator[comboBox2.Text], Communicator[comboBox3.Text], DTOC SAM[comboBox4.Text])
    - Sets : PostToHipChat(roles)
    - Sets : post.setRID(roomName)
  + Sends roles.ToString() to the selected HipChat room and closes the form
  + WebException will set the System.Clipboard with the roles in plain text to paste into HipChat
* ***checkRoles() : bool***
  + Checks the comboBox1-4
    - If blank returns false

### Purpose

BabyJellies.cs is used to post the DTOC roles into the selected HipChat room in the authorized format:

## Colors.cs

### Instance Variables

* List<Color> priColor
* List<Color> secColor

### Constructor

+ ***Colors()***

#### Default Settings

When the Color form is opened it will load with the following settings:

Sets : priColor

Sets : secColor

### Methods

* ***button1-10\_paint(object, PaintEventArgs) : void***
  + provides the logic for the background gradients on each button
* ***button1-10\_Load() : void***
  + initializes the PaintEventHandler(PaintEventArgs) class
* ***button1-10\_Click(object, EventArgs) : void***
  + Calls the Jellies.cs as a parent form and changes the background to the color set in the background of the button

### Purpose

For users to change the background color of the Jellyfish Status Tool form to a nice gradient.

# Class Descriptions

## PostToHipChat.cs

### Instance Variables

* Static string TOKEN
* Static string URL
* Static string ROOMS
* Static string mURL
* Static string notURL
* Static string getRoomURL : URL, ROOMS, TOKEN
* jellyOrRoles jr
* JellyfishStatusUpdate jelly
* Roles roles
* JObject jo
* String roomID
* List<string> roomNames
* List<string> ids
* String message

### Constructor(s)

***+*** ***PostToHipChat()***

***Used in only the initial form load to get the HipChat rooms***

***+ PostToHipChat(Roles)***

***Used to generate the roomID for posting Roles to HipChat***

#### Default Settings

* Empty Constructor
  + Initializes the List<string> roomName
  + getRooms() : void
* Overloaded Constructor : Roles
  + Sets : roles
  + Sets : jellyOrRoles.Roles
  + getRooms() : void

### Properties

* ***+ Roles***
  + Get : roles
  + Set : roles = value
* ***+ JR***
  + Get : jr
  + Set : jr = value
* ***+ RoomID***
  + Get : roomID
  + Set : roomID = value
* ***+ RoomName***
  + Get : roomName
  + Readonly
* ***+ IDS***
  + Get : ids
  + Readonly

### Methods

* ***setRID(string) : void***
  + uses string to search through the roomName List<string> using its property (RoomName) and find the index of the matching value
  + Sets : RoomID as the value of the element at the index located by the call above.
* ***getRooms() : void***
  + Makes HttpWebRequest GET request to the HipChat API.
    - Initializes the List<string> roomName and List<string> ids
    - Sets the two lists with the response from HipChat
      * The response is in a JSON format and is Parsed using JObject
      * The Parsed JObject is used to fill the two lists
        + roomName HipChat.name
        + ids HipChat.id

### Purpose

Helper class for Jellies and Roles

## JellyfishStatusUpdate.cs

### Instance Variables

* fullUpdateType type
* location local
* isResolved reso
* string incTitle
* string teamsEngaged
* string currStatus
* string busiImpact
* string techImpact
* string actWork
* string comWork
* string sandBox

### Constructor

***+ JellyfishStatusUpdate(string, string, string, string, string, string, string)***

#### Default Settings

When called the JellyfishStatusUpdate class will have the following settings

* Type = fullUpdateType.NONE
* Local = location.NONE
* Reso = isResolved.STATUS
* Instance variables (incTitle, teamsEngaged, currStatus, busiImpact, techImpact, actWork, comWork) will be set with the values from the constructor.
* sandbox will be set as empty

### Properties

* ***+ FullUpdateType***
  + Get : type
  + Set : type = value
* ***+ Local***
  + Get : local
  + Set : local = value
* ***+ Reso***
  + Get : reso
  + Set : reso = value
* ***+ IncTitle***
  + Get : incTitle
  + Set : incTitle = value
* ***+ TeamsEngaged***
  + Get : teamsEngaged
  + Set : teamsEngaged = value
* ***+ CurrStatus***
  + Get : currStatus
  + Set : currStatus = value
* ***+ BusiImpact***
  + Get : busiImpact
  + Set : busiImpact = value
* ***+ TechImpact***
  + Get : techImpact
  + Set : techImpact = value
* ***+ ActWork***
  + Get : actWork
  + Set : actWork = value
* ***+ ComWork***
  + Get : comWork
  + Set : comWork = value
* ***+ SandBox***
  + Get : sandBox
  + Set : sandBox = value

### Methods

* ***fixBrokeWorkaround() : string***
  + Returns the type of Status update using the FullUpdateType and isResolved property
* ***indiUpdate() : string***
  + Used when an individual field (Short Description, Current Status, Business Impact, Technical Impact, Active Workstreams, Completed Workstreams, The Sand Box) “post” button is pressed. Generates the title for the post to HipChat.
    - Determines the location via the Location property
* ***+ ToString() : string***
  + Generates the JSON string for the PostToHipChat.cs class.
    - Uses the fullUpdateType to determine the string that is generated
* ***+ errorString() : string***
  + In the event of a WebException, returns a text based string that is set to the System.Clipboard.

### Purpose

Helper class for the Jellies Form.

## Roles.cs

### Instance Variables

* String nameCordRole
* String nameEscaRole
* String nameCommRole
* String nameDSamRole

### Constructor

***+ Roles(string, string, string, string)***

#### Default settings

Instance Variables (nameCordRole, nameEscaRole, nameCommRole, nameDSamRole) will be set with the values from the constructor

### Properties

* ***+ NameCordRole***
  + Get : nameCordRole
  + Set : nameCordRole = value
* ***+ NameEscaRole***
  + Get : nameEscaRole
  + Set : nameEscaRole = value
* ***+ NameCommRole***
  + Get : nameCommRole
  + Set : nameCommRole = value
* ***+ NameDSamRole***
  + Get : nameDSamRole
  + Set : nameDSamRole = value

### Methods

* ***genJson(string) : string***
  + Generates the JSON string
* ***genKPI() : string***
  + Uses the Role properties to format the post for the genJson(string) method.
* ***+ ToString() : string***
  + Returns the string object generated by genJson(genKPI) method call.
* ***+ getRoles() : string***
  + Returns a plain text format of the roles posting in the event of a WebException

### Purpose

Helper class for the BabyJellies form

## SpellBox.cs

Imported class from WPF. I used this post to implement this code: <http://stackoverflow.com/questions/4024798/trying-to-use-the-c-sharp-spellcheck-class/4026132#4026132>

## Types.cs

Defines the enum(s) used in the helper classes

### Enums

* ***+fullUpdateType***
  + NONE - default
  + STATUS – used for Current Status, Business Impact and Technical Impact
  + WORKSTREAMS – used for Active and Completed Workstreams
  + BOTH – used for Status and Workstreams
* ***+location***
  + NONE – default
  + CUR\_STATUS – the Current Status field on the Jellies Form
  + BUS\_IMPACT – the Business Impact field on the Jellies Form
  + TECH\_IMPACT – the Technical Impact field on the Jellies Form
  + ACTIVE\_STREAMS – the Active Workstreams field on the Jellies Form
  + INACTIVE\_STREMS – the Completed Workstreams field on the Jellies Form
  + SAND\_BOX – the Sand Box field on the Jellies Form
  + INC\_TITLE – the Short Description on the Jellies Form
* ***+roles***
  + COORDINATOR
  + ESCALATOR
  + COMMUNICATOR
  + DTOC\_SAM
* ***+isResolved***
  + RESOLVED – used when the Resolved checkbox on the Jellies Form is checked
  + FINAL – used when the Final checkbox on the Jellies Form is checked
  + STATUS – default setting
* ***+jellyOrRoles : deprecated***

### Purpose

Used to control the various states of the Jellyfish Status Tool