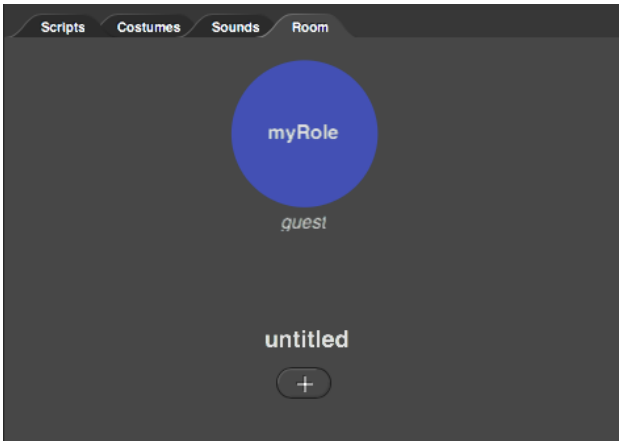
NetsBlox Lesson: Introduction to Messages
Chatroom

Remote Procedure Calls (RPC) provide one way to write a distributed program. A program using an RPC is distributed because part of it runs on your computer (inside your web browser), but the actual RPC runs on a different computer, the NetsBlox server. In fact, many of those RPCs will then request data from other sources, for example, Google provides mapping data. That means that even more computers are involved in a simple RPC call.

Today we'll look at another way of creating a distributed program using *message passing*, a technique widely used to share data between program running on different computers. Message passing enables you to send data to another NetsBlox program running on a different computer. In some sense, messages are similar to events we have seen before (remember the `broadcast` and the `When I receive blocks`?). There are two differences though: messages can contain data and they can be sent to NetsBlox programs running on different computers not just different sprites within the same program.

When you send an email to somebody, what is it that you must know? That's right, the email address. The address uniquely identifies your recipient: no two people can have the same address. Just like with phone numbers, postal mailing addresses or website url-s, you need a globally unique identifier. That works the same way with NetsBlox. If we want to send a message to another program, we have to make sure that we can uniquely identify our recipient. To understand how NetsBlox message addressing works, we need to introduce a new concept.

You might have seen the Room tab next to Scripts, Costumes and Sounds. If we select it, you'll see something like this:



Every NetsBlox project has a single Room that has one or more Roles. The Role is the concept NetsBlox uses to allow multiple subprojects within one project. A single NetsBlox project can have multiple Roles, that is, multiple subprojects, each of which can run on separate computers. For example, if we create a multi-play game like Tic Tac Toe, Chess or Battlefield,

we need two players. Each of these players is associated with a Role. For example, Tic Tac Toe would have two Roles named X and O, while chess might use the names black and white for its two Roles. A poker game would typically have more than two Roles, one for each of the players participating. Each of these Roles has its own sprites, costumes, scripts, etc. In other words, a Room is a way to group together NetsBlox projects that jointly implement a multi-player game or other distributed program.

Back to messages and the problem we raised: how to identify the recipient of a message? Or in computer networking terminology: how to address a message? In NetsBlox, the addressee of any message is a Role.

In fact, it is really easy to send messages to Roles within the same Room. Let's create our first messaging application! When you start learning a new computer language, the first program you write is typically one that prints "Hello World" on the screen. Our first messaging app is going to be a distributed version of Hello World. One Role will send the text "Hello World" to another Role, which, in turn, will print it on the stage.

Distributed Hello World Project

Let's create a new project, go to the Network tab, drag in the rectangular **`send msg`** block, and select the only option from the pull-down menu in the middle, called message. This is what you should see:

Messages have different types. The message type consists of a name and the list of fields, that is, the list of data items the message carries. NetsBlox comes with one built-in message type called **`message`** and that is what we picked from the menu. It has one field called **`msg`**. For this app, it will work just fine. Later we'll see how to create new message types.

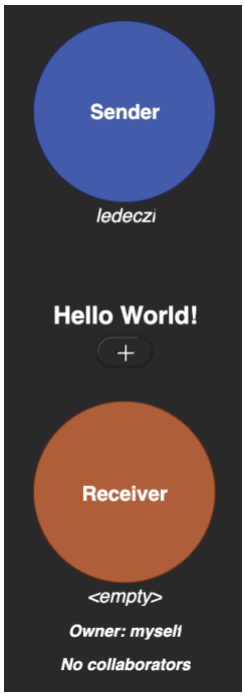
Back to our **`send msg`** block. The only data field this message type has is called **`msg`** and this is where we will type in our message: Hello World!

The rightmost pull-down menu is where we will put the address of the message. Since we have not created another Role, let's leave it alone for the time being.

Instead, let's create a second Role, the one that will receive our message. Click on the Room tab. We see the single Role called "myRole" and underneath it the name of the project called "untitled". Let's click on the Role name (make sure it is the name and not the blue background). Let's type the new name: "Sender". This Role will be used to send a message to the other Role.

We can also click the name untitled and type in a new name for our project. I'll use "Hello World!"

Now let's create that second Role, shall we? Click the plus sign in the middle and type in the name "Receiver". This is what you should see (except your username should show up, not mine, ledeczi).

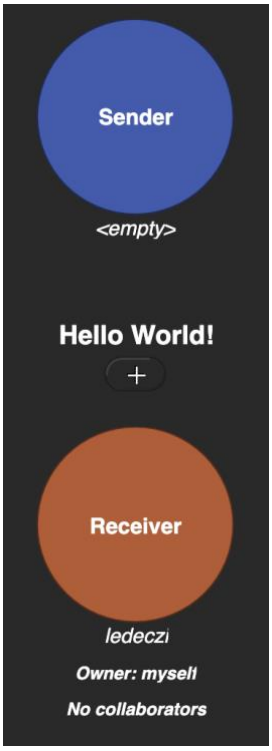


We have two Roles: Sender and Receiver. The name of the project is "Hello World!" We are occupying, that is, working with, the Role Sender and currently nobody is occupying the Role Receiver (indicated by the text <empty> under the brownish circle). Finally, we own this project and there are no collaborators.

NetsBlox allows multiple users to work on a project together, that is, to collaborate, very much like how Google Docs work. We won't go into that today.

Ready to write the code for the Receiver Role? Click the brownish background of the circle and select the "Move to" option. We are asking NetsBlox to switch Roles for us. When you switch Roles, any unsaved changes are lost, so NetsBlox asks you to Save your Role and you should click Yes.

You may or may not have noticed that only a small change occurred on the screen. The Sender Role is now showing <empty> and the Receiver Role has your username under it. Here is what I see:



Let's click on the Scripts tab. There is nothing there?! It should not be surprising though: we have created a brand-new Role and just like a new project, it starts out blank. A new Role is a new subproject. We have to create the sprites and write the scripts for them.

So, let's do just that! Go to the Network tab and drag in the `when I receive block` and select the only option in the pull-down menu, **`message`**.

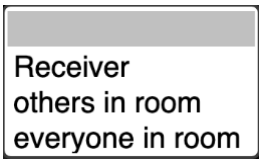
As you might have guessed, the pull-down menu lists all the available message types. Since we have not created any, it only shows the single built-in message type, called appropriately enough, **`message`**. And its only data field, called **`msg`**, shows up in the block as a variable!

All we have to do now is display it by using a `say` block (under the Looks tab). Simply drag the **`msg`** variable from the `when I receive block` and drop it in the `say` block. This is the entire script of the Receive Role:

Let's save our work. Notice that in the File menu, now it says Save Role as opposed to just Save. Once you have multiple Roles, you can only save the one you are working on.

Our Receive Role is ready, but we left one more thing to add in our Sender Role. So, let's go back. Click the Room tab and click the blue background of the Sender Role and pick the "Move to" option. If you have not saved the Receiver Role, click Yes when NetsBlox offers you to save the Role before switching.

Click the Scripts tab and let's check out the `send msg` block we added previously. We left the address blank. Click the pull-down menu and this is what you should see:

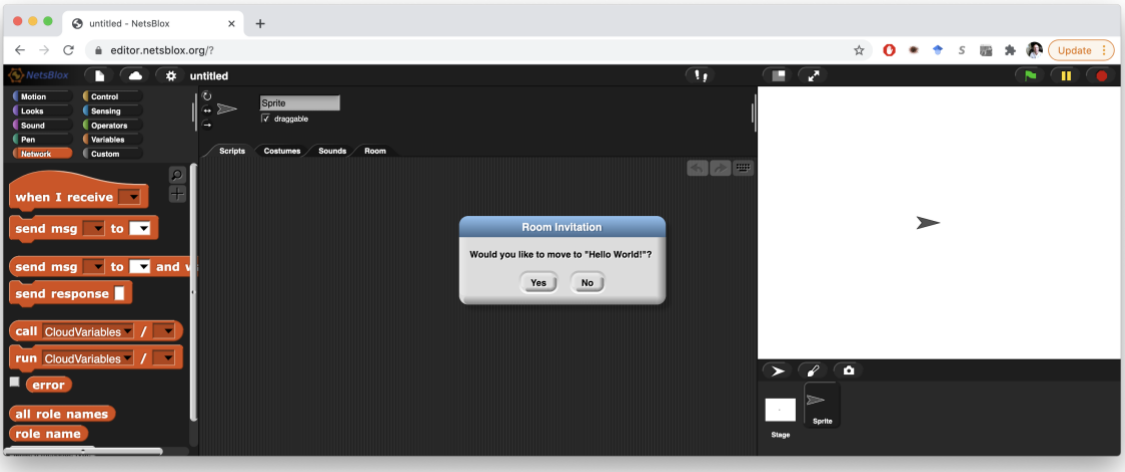


The first option is the only other Role in the project, Receiver. The "others in room" option means every other Role. Since there is only one other Role, this is equivalent to picking the Receiver Role in our case. The "everyone in room" option means every Role including the sender itself. Let's pick the Receiver option.

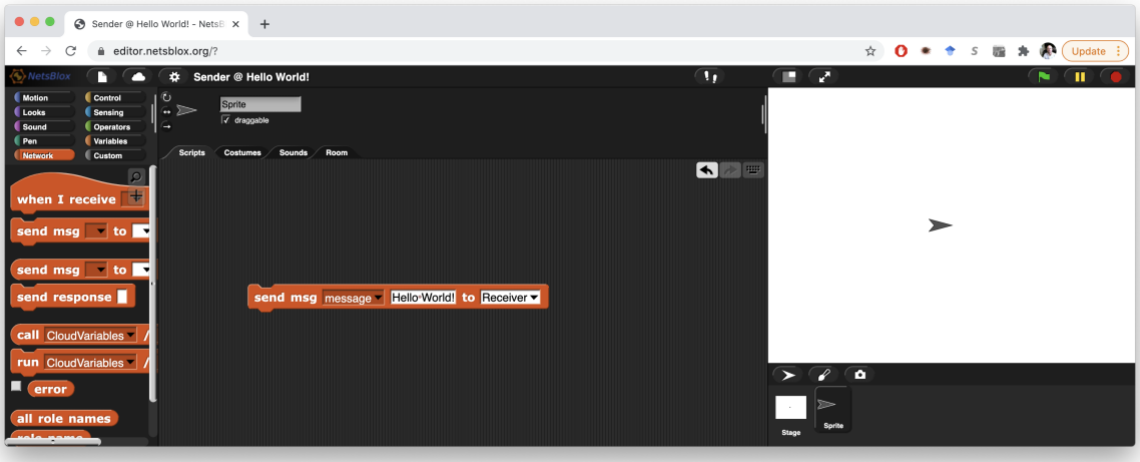
We are ready with our distributed Hello World app! It only took a grand total of three blocks of code! The question is: how can we test it? Don't we need two computers? It turns out that you can test your multi-role distributed NetsBlox programs in separate browser windows/tabs. Browser tabs are isolated from each other, so it is actually a perfectly valid test.

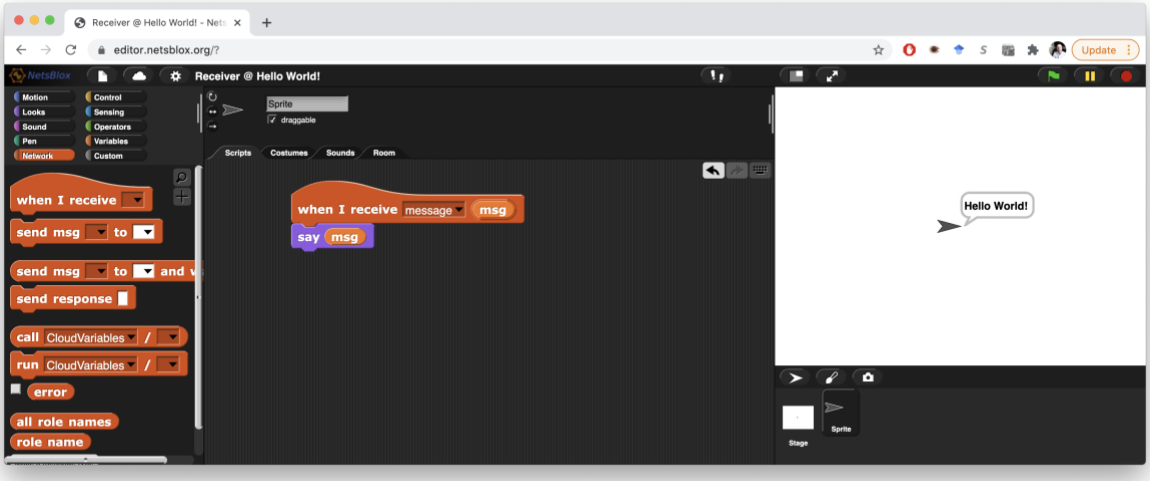
So, let's create a new browser tab and open NetsBlox in it as well. Position the two browser windows so that you can see them both on your screen. (If the browser tabs live in the same window, only the one that is visible is active. That means that a NetsBlox project will not run if another tab in the same window is on top.) In the original browser where we have the Hello World app open, go to the Room tab and click on the background of the receiver Role:

Select the Invite User option. The new window that pops up shows all currently active NetsBlox users, but the very first one in the list says: myself. Click that. In the second browser window, you should see an invitation to join the Hello World project:



Click Yes and you will find yourself in the receiver Role of the Hello World project once it is loaded. In the first browser tab, click the `send msg` block. You should see the sprite in the second browser window say Hello World like this:



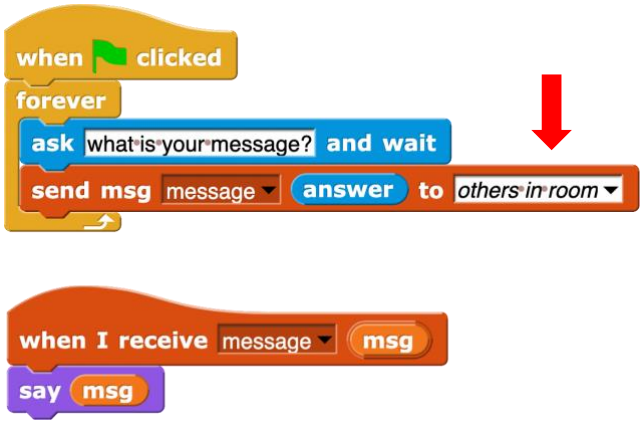


If you want to experiment a little bit, change the hello World text in the `send msg` block to say something else, like “NetsBlox is really cool!!!” Click the block again and you should see the text pop up on the other side.

So, our first distributed application that uses message passing is complete! You may think that is fairly limited. Well, then let’s turn it into a real texting app, shall we?

Texting Project

The idea is to keep asking the user to type in a message and then sending it to every other Role in the project. Let’s create a new blank project. Let’s add the sending code like before, but also the `when I receive msg` block so that every Role will be both a sender and a receiver just like in a normal texting program. This is how the script should look like:



Notice how we used the “others in room” address in the `send msg` block this time. This way, every Role can have the exact same script since we are not using specific Role names!

Now let’s create one or more additional Roles. Go to the Room tab, rename the current Role to something like “Person 1” by clicking on the original name “myRole.” You can also rename the project from untitled to something more meaningful. Here comes the trick: click on the blue background of the Role and select Duplicate. This will make an identical copy of the Role. Feel free to rename the new Role to something like “Person 2.” If you switch to this Role, you’ll see that the scripts are all the same as in the original Role.

Let’s test the project the same way we did the Hello World one. Create an additional browser window, open NetsBlox and invite “myself” to the currently empty Role (Person 1 or Person 2). Click the green flag in both browser windows and you can send message back and forth between the two NetsBlox windows.

If you have two computers handy, you can log in on both, invite yourself from one and you can use the app to send messages across the computers.

You can also invite your friends to use the app. They need to open NetsBlox on their computer, log in, wait for your invite and then accept it.

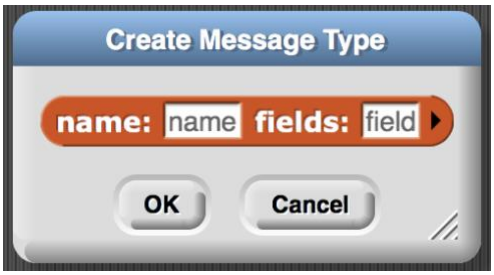
If you did not quite finish your project, here it is:
https://editor.netsblox.org/?action=present&Username=ledeczi&ProjectName=VDN-Texting&

Extending the Project

It is easy to extend the project to have more than two people participate. You can create additional Roles by duplicating one and inviting multiple friends. In that case, the current downside of our implementation is that you do not know who sent which message. How can we fix that?

We should send a message that has two fields: the sender name and the actual message text. For that, we need to define a new message type.

How do we define new message types in NetsBlox? In the Network tab, there is a gray button called `Make a message type`. When you click it, this window pops up:



We type in a new name, let's say "chat" for the name of the new message type. Click the arrow to add a second field. We can name the first field "sender" and the second "text".

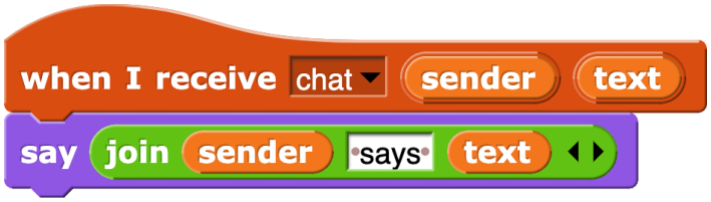
Press OK. To see what message types we have in our project, we can click on the Room tab and there will be listed in the top left corner. If you click on one, it shows its fields too:

Or if we pull down the menu of the `when I receive msg` block, we see all available message types too:

If we select the new type, “chat” we get both fields as variables as expected:

Well, the rest is pretty easy for such a seasoned NestBlox programmer as you. We modify the sender script so that it asks the user for their name, save it in a new variable and then modify the `send msg` block to use the chat message type. Finally, we insert the name (`my name`) as the first field and move the actual message (`answer`) into the second:

The receiver script is even easier:



We pick the chat message type and display both the sender and the message using a `join` block. Now you can keep duplicating the Role to get as many people participate as you need.

Here is this version:
https://editor.netsblox.org/?action=present&Username=ledeczi&ProjectName=CSF-Texting&

The remaining limitation is that it only displays the last message. Here is a more advanced version that stores the last few messages in a list and displays them all on the stage using a custom block. We won't describe it here, but feel free to explore it and modify it as you see fit:

https://editor.netsblox.org/?action=present&Username=ledeczi&ProjectName=Texter&