The HypeDyn Hypertext Fiction Editor
Tutorial 4: Number Facts

Contents

1 Introduction **1**

2 Getting started **2**

3 Creating a number fact **3**
3.1 Creating the fact 3
3.2 Creating the alternative endings 3
3.3 Setting the number fact 4
3.4 Displaying a number fact using alternative text 4

4 Comparing number facts **5**
4.1 Adding the first rule 5
4.2 Testing the first rule 6
4.3 Adding the second rule 6

5 Using random numbers **7**
5.1 Setting the random number in a node rule 7
5.2 Setting the random number in a link rule 8
5.3 Adding a third ending 8

6 Using a counter **10**
6.1 Creating the counter 10
6.2 Incrementing the counter 11
6.3 Using the counter to unlock the side path 12

7 Next steps **12**

8 Conclusion **13**

1 Introduction

In this tutorial, we will introduce "number facts", which are similar to text facts and true/false facts but can be used to store *numbers*. Number facts can be

updated to a specific number, to the value stored in another number fact, or to a random number. You can also perform simple math on number facts, and use number facts in conditions. We will be creating a variation of the "Little Red Riding Hood" hypertext fiction which you created in tutorial 1. We will change the story such that the choice of ending is determined by a number fact, which can either be preset by the author or chosen randomly when the story is being read. We will also change the side path ("Hood details") such that it is unlocked, not after the first reading, but instead after the *second* reading, through the use of a counter.

The nodes and facts in the final story are shown in Figure 1.

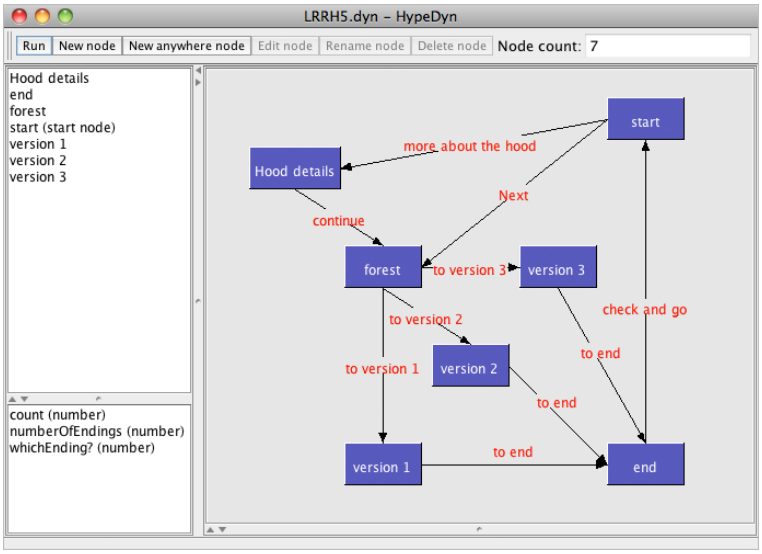

Figure 1: *The completed "Little Red Riding Hood" story.*

Note: HypeDyn is a work-in-progress, so there are some features that are still not completed, and there may be bugs. If you encounter any errors, please report them as bugs on our Launchpad site: `https://launchpad.net/hypedyn`.

2 Getting started

First, open HypeDyn by double-clicking on the file **HypeDyn.exe** (in Windows) or **HypeDyn.app** (in MacOS).

We will continue from the story that you created in tutorial 1. If you don't have the story, you can start from the file LRRH.DYN in the EXAMPLES folder. HypeDyn files always end with a **.dyn** extension. Once you've opened LRRH.DYN, save the file under a different name, such as LRRH5.DYN.

3 Creating a number fact

Number facts are similar to the text facts and true/false facts which were introduced in Tutorial 3. Instead of keeping track of text or boolean values, number facts are *integers*, meaning they can be set to any positive or negative "natural" or "whole" number ie. ...-3, -2, -1, 0, 1, 2, 3 ..., but not *real* numbers with a decimal place. Number facts can be tested in conditions, and set in actions.

The first feature we will introduce is the use of a number fact in a condition. As with other conditions, this allows you to determine which actions should be triggered, either when a node is entered or when a link is clicked.

3.1 Creating the fact

We will begin by creating the number fact.

1. In the main HypeDyn window, go to the *Fact* menu, and pick the *New* menu item, and the *Number* submenu item.

2. Name the fact "whichEnding?", and click "Ok". As with other facts, your new fact will appear in the fact list in the lower left corner of the main window.

We now have a *number fact* which we can use to keep track of numeric values.

3.2 Creating the alternative endings

What we want to do with our "whichEnding?" number fact is use it to determine which of two possible endings will be visited when the reader clicks on "next" in the "forest" node. Currently it leads to the node "end". We will now create 2 new nodes, "Version 1" and "Version 2", which will be shown depending on the value of "whichEnding?". Create two nodes:

1. *Name*: Version 1
   Content:

   > Quickly the wolf ran ahead to Grandma's house, swallowed Grandma whole, and disguised himself as the poor old lady. When Red arrived, he finished her off too. Yum!
   >
   > next

2. *Name*: Version 2
   Content:

   > Quickly the wolf ran ahead to Grandma's house. Unfortunately for the wolf, Grandma's friend the Woodcutter had stopped by for tea, and he finished the wolf off with one swipe of his axe.

Poor wolf!

next

3. Add a link from the "next" text in each node to the "end" node.

3.3 Setting the number fact

We want to have the reader go from the "next" link to "Version 1" if "whichEnding?" is equal to 1, and to "Version 2" if "whichEnding?" is equal to 2. Before we can do this, however, we need to set the value of "whichEnding?" As with other types of facts, a number fact can be updated in either a node rule or a link rule. We will use the node rule in the "forest" node to set the value.

1. Edit the "forest" node, and edit its node rule.
2. Add a rule, and name the rule "choose ending".
3. Edit the rule, and add an "update fact" action.
4. In the new action, choose "Number" for the fact type, and choose fact "whichEnding?" for the fact.
5. Notice that there are several options in the "using" pulldown menu. Number facts can be updated based on a number that the author inputs directly (the *input* option), much like specifying alternative text when updating text on a link. They can also be updated using another fact, using a mathematical expression, or using a random number

   For now, we want to specify the number directly, so choose "input". In the input field, specify "1".

The rule we've created will set the "whichEnding?" fact to 1 when the reader enters the "forest" node.

3.4 Displaying a number fact using alternative text

To check whether our new rule is working, we can temporarily add a link which we will use to display the current value of "whichEnding?" This technique is a useful way to debug your stories when using number facts.

1. At the end of the "forest" node, add the text "whichEnding? = dummy". We will replace the text "dummy" with the current value of whichEnding?
2. Create a link on the text "dummy", name the link "debugging", and edit the link.
3. Add a rule to the link, and name the rule "debugging".

4. Add an "update text using" action to the rule, and choose update text using "number fact". In the pulldown menu, choose the fact "whichEnding?"

5. Close the rule editor and the link editor.

6. Now run the story. Notice that the text "dummy" is replaced with the value "1".

7. Edit the "forest" node's node rules, go into the "choose ending" rule, and change the "update fact using" action so that the value is 2. Close the rule and node rules editors.

8. Run the story again. You should see that the value displayed in the forest node is now "2".

Now that we've created our number fact, and have set it to a specific value, we can use this number fact to decide which ending to show.

4 Comparing number facts

We will now add a condition to the "to the end" link in the "forest" node, which checks the value of the "whichEnding?" fact. If the value is 1, the link will go to "Version 1", and if the value is 2, the link will go to "Version 2".

4.1 Adding the first rule

We'll begin by adding the rule which takes us to "Version 1".

1. Edit the "forest" node, and then edit the "to the end" link. You should see the "to the end" rule which we created in tutorial 1.

2. Edit the "to the end" rule. It currently contains one action, "follow link to end". Rename the rule "to version 1".

3. Now change the action to "follow link to Version 1". This is the action we will use if "whichEnding?" is 1.

4. Add a condition. Choose "Number Fact" as the type of condition.

5. Notice that after changing the condition to a number fact condition, the condition changes to show two pulldown menus and a text input field. The first pulldown menu contains $=$, $<$, $>$, $\leq$ and $\geq$. This is the *comparator* which will be used to compare the chosen number fact (in this case "whichEnding?") with the right-hand side of the condition.

   For now, leave the comparator as "=".

6. To the right of the comparator is another pulldown menu, which has two options: input and fact. Choosing "Input" lets you enter a specific number for the comparison, whereas choosing "Fact" lets you compare against another fact. Leave the choice as "Input".

7. In the text entry field, enter "1". This means that the actions listed in the rule will be triggered when "whichEnding?" is equal to 1.

8. Close the rule editor.

4.2 Testing the first rule

At this point, we've created the rule which will take the reader to node "Version 1" if the fact "whichEnding?" is equal to 1. Recall that we have currently set the node rule in the "forest" node to set "whichEnding?" to 2. What will happen if we run the story now?

1. Run the story.

2. Notice that the link on the text "next" in "forest" is disabled – this is because it contains rule with a "follow link to" action, but that rule's conditions are not satisfied.

3. Now edit the "choose ending" node rule in the "forest" node, and change the "update fact" action so that "whichEnding?" is set to 1.

4. Run the story again. This time, the link is enabled. Click on the link, and you will be taken to node "Version 1".

4.3 Adding the second rule

Now we need to add the second rule, which will take the reader to node "Version 2" if "whichEnding?" is 2.

1. Edit the "to the end" link in the "forest" node.

2. We are going to add a second rule, which should only ever be triggered if the first rule, "to version 1", was **not** satisfied. To make sure that HypeDyn stops checking rules after a given rule is satisfied, you can check the "Stop if true" option on a rule. Do this now.

3. Add a rule, and name the rule "to version 2".

4. Add an action "follow link to Version 2".

5. We only want this rule to be triggered if "whichEnding?" is set to 2, so we need to add a condition to check this.

   Add a condition to the rule, and set it to "Number fact whichEnding? = input 2".

6. Run the story. When you get to "forest" and click on the "next" link, you should go to "Version 1".

7. Now edit the node rule for "forest", and change the "choose ending" rule so that "whichEnding?" is set to 2.

8. Run the story again. This time, the "next" link should take you to "Version 2".

We have now successfully created a link which goes to different destinations based on a number fact.

5 Using random numbers

Although we've created a set of rules which use number facts to determine which node to visit next, this isn't really anything which we couldn't have done with, for example, a true/false fact. One way to make this more interesting, and to show what can be done with number facts, is to set the fact "whichEnding?" to a *random number*.

5.1 Setting the random number in a node rule

We will now create a rule which sets "whichEnding?" to a random number. There are several places where we could put this rule. To start with, we will change the "choose ending" node rule in the "forest" node to update "whichEnding?" to a random number, rather than a number which we decided when writing the story.

1. Edit the "choose ending" rule in the node rules for the "forest" node.

2. There is currently one action: "update fact Number whichEnding? using Input 2". Change the "using" pulldown to "random". Notice that there are now two input fields to the right of "random". These input fields allow you to choose the lower and upper range of the random number which will used to update the number fact. These bounds can be specified either using Input or a Fact. For now, leave both pulldowns as Input.

3. We want a random number between 1 and 2, since there are 2 versions of the ending. So enter "1" in the first input field, and "2" in the second input field.

4. Click "Ok" to close the rule editor, and "Close" to close the node rules editor. Now run the story. When you enter the "forest" node, the value displayed at the bottom of the node should be either 1 or 2.

5. To convince yourself that the value is being chosen randomly, click on the "back" button, and then click on the "forest" link again. Do this a few times. Notice that sometimes the same number appears two or more times

in a row – this is because the number is chosen randomly, without regard to what the previous value was.

5.2 Setting the random number in a link rule

Note that although we put the update fact action which sets "whichEnding?" on the node rule for the "forest" node, instead we could have put the action on the "next" link itself. One advantage of having the action on the node rule is we can see the value of the "whichEnding?" fact (on the "dummy" link in the node's text), which helps with debugging. However, it can sometimes be useful to put actions such as this on a link. To show how this is done, we will now put the same action that we put in the node rule into a rule on the "next" link.

1. Edit the "next" link in the "forest" node.

2. There are already two rules on the link: "to version 1" and "to version 2". We want to update the "whichEnding?" fact, and *then* check the value of the fact. Remember that HypeDyn checks, and then triggers, rules in the order they are listed in the rules editor. This means that we will have to put our new rule before the two existing rules.

   Add a new rule. Notice that it appears at the end of the list of rules. We need to move it up so that it is before "to version 1".

3. Make sure your new rule is selected (it should be highlighted in blue". Now click on the "up" button. The rule should have moved up, so that it is now before "to version 2", but still after "to version 1".

4. Click "up" again. The new rule should now be first in the list, before "to version 1".

5. Edit the new rule, and name it "randomize".

6. Add an "update fact" action, set it to "update fact Number using Random between Input 1 and Input 2".

7. Click "Ok" to close the rule editor, and then click "Close" to close the link rules editor.

8. Try running the story, and clicking the "next" link several times. Notice that the destination the link goes to is not always the same as what you would expect based on the number displayed in the "forest" node. This is because the "next" link's first action is again randomizing the value in "whichEnding?"

5.3 Adding a third ending

Note that it is not very good form to have two different places where "whichEnding?" is randomized, as you may lose track of where the randomization is taking

place. There is also another problem – if we add another ending, we now need to remember to change the upper bound for the random number from two to three in two different places!

Sometimes there are good reasons for having randomization such as this happening in several places. For example, there may be several different nodes which lead to the endings, each of which need to do some randomization based on the number of endings. One way to improve the situation is to use a *fact* for the upper bound, rather than a number we have typed in two different places. Then we can just change the value of the fact, and everywhere the fact is used, we'll get the correct value. We will do that now.

1. First, we'll make a third version of the ending. Create a new node, and name it "Version 3".

2. Enter the following text in the node:

   > Fortunately for Red, the wolf got lost on the way to Grandma's house.
   >
   > next

3. Add a link from the text "next" to the "end" node.

4. Now we will add a new number fact to keep track of how many endings there are. Create a new number fact, and name it "numberOfEndings".

5. We need to update this fact to "3". The best place to do this is at the start of the story.

   Edit the "start" node, and edit the node rules for this node.

6. Add a new rule, and call the rule "initialize". This is where we will set the value of "numberOfEndings".

7. Add an "update fact" action to the rule, and set it as follows: "update fact Number numberOfEndings using Input 3".

8. Now we need to make use of this new fact in our two rules where we randomize "whichEnding?".

   Edit the "forest" node, and edit its node rules. Edit the rule "choose ending". In the first action, change the second "Input" to "Fact". This is the upper bound of our random number. Choose the fact "numberOfEndings".

9. Now do the same for the "randomize" rule in the "to the end" link's rules.

10. Finally, we need to add a rule to the "to the end" link to take the reader to node "Version 3" if "whichEnding?" is equal to 3.

    Add a rule to the "to the end" link's rules, and name it "to version 3". Make sure the rule is at the end of the list of rules.

11. Add the condition “Number Fact whichEnding? = Input 3”.

12. Add the action “follow link to Version 3”.

Now try running the story. You should be able to follow the “next” link in the “forest” node, and have it randomly take you to one of the three endings.

In this section, we introduced the use of a random number fact to create a link which randomly takes the reader to different destinations. This is a powerful procedural technique. It can, however, be problematic if misused. If choices the reader is making result in random outcomes, the reader may begin to doubt that her choices have any real impact on the story. This may or may not be what you intend as an author.

6 Using a counter

In this section, we will introduce another way in which authors can use number facts to make their stories more procedural. Here, we will update facts based on math. We will demonstrate this by implementing a *counter* which unlocks a side path in the story after the reader has reread the story a certain number of times. To do this, we will create a new number fact, named “counter”. Each time the reader clicks on the “restart” link from the “end” node back to the “start” node, we will add 1 to “counter”. Then, in the “more about the hood” link, instead of checking whether or not the “end” node has been visited, we will check the value of “counter”, and enable the link when the target value has been reached.

6.1 Creating the counter

First we will create a number fact to act as our counter.

1. Create a new number fact, and name it “counter”. We’ll use this to keep track of the number of times the reader has gone back to the start of the story from the “end” node.

2. So that we can see how the counter is changing, we will use a technique similar to what we did with the “whichEnding?” fact to display the value of “counter”.

   Edit the “end” node, and add the text “counter=placeholder” between the text “*** The End ***” and the link “back to start”. Add a link on the text “placeholder”, name the link “show counter”, and add a rule which has the action “update text using number fact counter”. This will show us the value of “counter” when we test our story.

3. Now run the story. When you get to the “end” node, notice that the value shown for “counter” is “0”.

Number facts are always automatically updated to zero when the story is started. In this case this is what we want, as the reader has initially not gone back to the start of the story.

6.2 Incrementing the counter

When the reader goes back to the start the first time, we will add 1 to the current value of counter, updating counter to 1.

1. Edit the "restart" link in the "end" node.

2. There is already one rule, "restart", which was added in Tutorial 1. This rule contains one action, "follow link to start", which takes the reader back to the "start" node. In addition, we want to increment the counter fact when the reader goes back to the "start" node.

   Add an "update fact" action to the "restart" rule. Set the type of update to "Number", select Fact "counter", and set the "using" pulldown menu to "Math".

3. After you choose "Math" for the "using" pulldown menu, you will see an "Input" pulldown menu, a "+" pulldown menu, and a second "Input" pulldown menu. These menus let you specify the *math expression* which will be triggered for this action.

   HypeDyn lets you update number facts using simple math expressions. Each one consists of two values (either an Input or a Fact), and an *operator*: +, -, x, / (div) and % (mod). Because number facts are whole numbers, the "/" operator is actually "div", which performs the division but discards the remainder. The "%" operator is called "mod", which performs the division and gives you the remainder.

   We want to add 1 to the current value of "counter". To do this, choose "Fact" for the first value. You will see the text field is replaced by a pulldown menu containing a list of number facts. Choose "counter".

4. For the second value, we want to add 1, so leave the second value's pulldown menu as "Input", and enter "1" into the text field.

5. Finally, we want to perform addition on the two values, so leave the operator pulldown showing "+".

6. Click "Ok", and then close the link rules editor.

Try running the story now. Click through to the "end" node, and then read the story a second time by following the "back to start" link. When you reach the "end" node a second time, you should see the text "counter=1". Try rereading several times, and you'll see the counter increase by 1 each time.

6.3 Using the counter to unlock the side path

We can now make use of this counter to unlock the "more about the hood" side path after the reader has gone through the story twice, rather than just once.

1. Edit the "more about the hood" link in the "start" node, and edit the "more about the hood" rule. This rule was added in Tutorial 1. It contains one condition, "Node end visited", and one action, "follow link to Hood Details".

2. We want to retain the existing action, but instead of triggering the rule if the "end" node has been visited, we want to trigger it only if the "end" node has been visited two times. To do this, we will use our "counter" fact, and use a number fact condition to check the value of the fact.

   Change the type of the condition to "Number Fact", and choose fact "counter".

3. We want to check whether the reader has visited the "end" node twice, so set the comparator pulldown menu to show "=", and make sure the type pulldown menu for the right-hand side is set to "Input". Then type the value "2" in the text entry field.

Now try running the story. Read through once, and then go back to the "start" node by clicking on the "back to start" link in the "end" node. Notice that the link on the text "red hood" is not clickable. Now continue to read, and once again to back to the start. This time, the second time you've gone back to the start, the "red hood" link *is* clickable.

What do you think will happen if you read through again, and then go back to the start a *third* time? Try it. Why isn't the link clickable the third time?

Remember, we set the condition on the link to be "Number fact counter = 2". This means that the link is clickable when "counter" is *exactly* equal to 2. This is fine if we want the link to only be available on the second rereading. If, however, we want it to be unlocked for every subsequent rereading, we should use "$\geq$" instead of "=". Try this, and then read the story again. This time, you should see the "red hood" link is clickable for the second and subsequent rereadings.

7 Next steps

We have modified the original story from Tutorial 1, LRRH.DYN, by adding random selection of endings, and by adding a counter which both unlocks the side path on the second rereading. The completed version of this story can be found in the file LRRH5.DYN.

There are several things that you could try to enhance the story. For example, you limit the number of rereadings to three, making sure that once you prevent the reader from rereading you let them know why. You could also make

sure that the reader doesn't get the same random ending twice in a row. One way of adding these enhancements can be found in LRRH6.DYN.

8 Conclusion

In this tutorial, we have created a simple hypertext fiction which makes use of randomization and a counter to procedurally alter the story which the reader encounters. This provides the basis for much more procedural approaches to creating interactive stories. See what you can think of, and then try it!