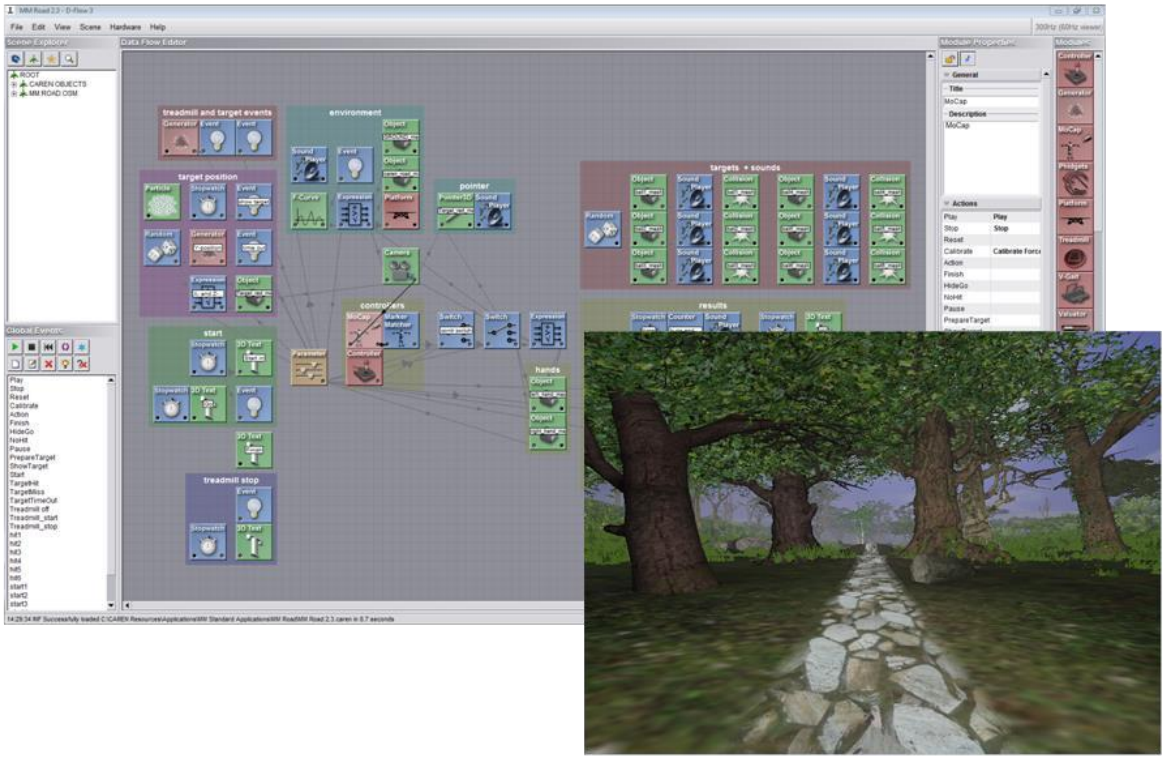
D-Flow

Training Syllabus



1 Preface

D-Flow is a software system designed for the development of interactive and immersive virtual reality applications, for the purpose of clinical research and rehabilitation. A key concept of the D-Flow software system is that the subject is regarded as an integral part of a real-time feedback loop, in which multi-sensory input devices measure the behavior of the subject, while output devices return motor-sensory, visual and auditory feedback to the subject. The D-Flow software system allows an operator to define feedback strategies through a flexible and extensible application development framework, based on visual programming.

The D-Flow Training Syllabus was designed by Motek to provide new D-Flow users with the knowledge they need to start using the software. This syllabus contains all the information about the basics of D-Flow.
We hope you enjoy discovering the possibilities of D-Flow.

Website:
https://www.motekmedical.com

Support:
clinical.applications@dih.com

Knowledge platform
https://knowledge.motekmedical.com/

User Forum:
https://www.linkedin.com/groups/9079702/

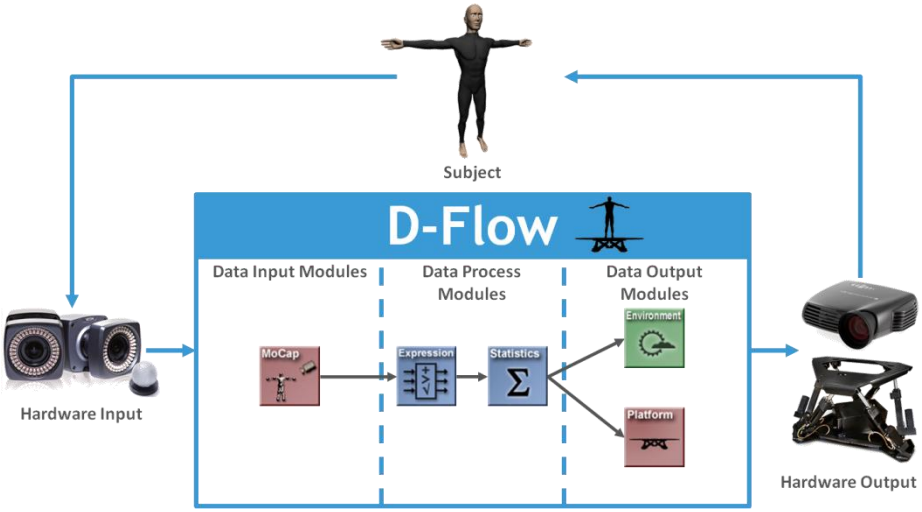
2 Contents

3 Basic D-Flow Training

This part of training is intended to provide the trainee with the rationale behind the software, an overview of the different aspects of D-Flow, a basic understanding of how to use D-Flow and how to build small applications. For more information on the software, you are advised to check the Help menu in D-Flow which can be opened by pressing F1 when the software is open.

3.1 D-Flow software

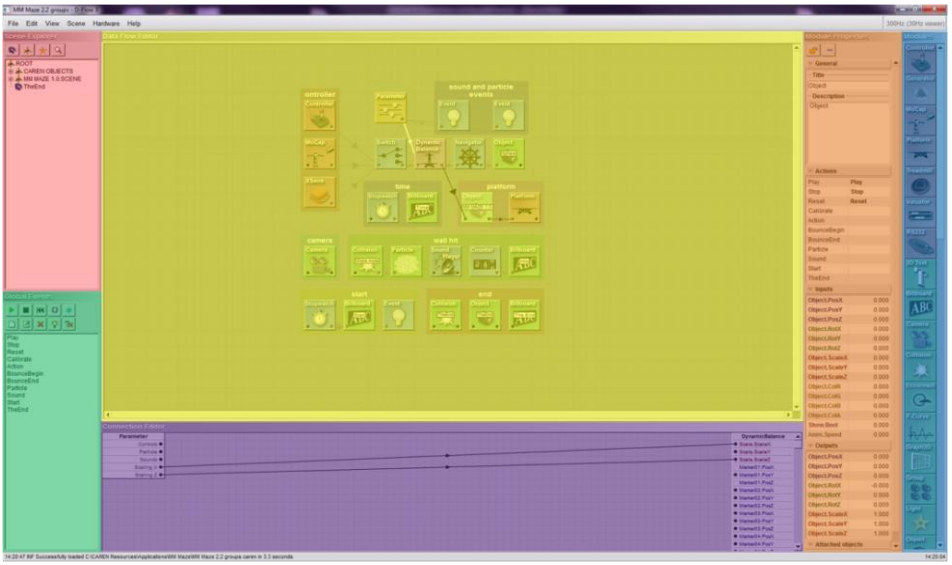
D-Flow is a software system designed for the development of interactive and immersive virtual reality applications, for the purpose of clinical research and rehabilitation. A key concept of the D-Flow software system is that the subject is regarded as an integral part of a real-time feedback loop, in which multi-sensory input devices measure the behavior of the subject, while output devices return motor sensory, visual and auditory feedback to the subject. The D-Flow software system allows an operator to define feedback strategies through a flexible and extensible application development framework, based on visual programming.



The D-Flow software system is a combination of the following components: a layer for integrated hardware communication, a multi-display rendering system, and a modular application development framework based on visual programming. Individually, these components are not new; similar components have been developed and put to use in many virtual reality, machine control or software development applications. The distinctive quality of D-Flow is that it combines these components into a single system, with a specific focus on clinical research and rehabilitation.

3.2 Overview user interface

An overview of the D-Flow Editor window is displayed below. Separate sections exist sections for organizing modules, editing connections, and assigning global events to module actions. These are shown by the different colours in the image below and are discussed below.



3.2.1 Data Flow Editor (yellow)

The Data Flow Editor shows all used modules and the connections between the modules. This section gives an overview of the currently open application and shows the data flow created between modules (the wires).

3.2.2 Modules section (blue)

The Modules section contains all available modules that can be used to create the D-flow applications. Each module is designed for a specific task in an application. For more information about the modules please refer to the modules section in the Help menu.

3.2.3 Scene Explorer section (red)

The Scene Explorer shows a hierarchical overview of all elements in the 3D scene. This includes among others lights and objects. Four buttons are available in the scene explorer:

	The 'Add Object' button is used to add a new stock object to the scene.
	The 'Add Scene' button is used to load a scene into the application.
	The 'Add Light' button is used to add a new light source to the scene.
	The 'Focus' selection in DRS button is used to focus the view of the DRS window on the selected object or light.

3.2.4 Global Events section (green)

The Global Events section is used to manage the global events of the application. This section contains a list of all available global events, the controls to create, modify, test and delete events and the play controls as seen in the Runtime Console.

	Play
	Stop
	Reset
	Calibrate
	Action

The play controls in the Global Events section provide the standard play controls that are also available in the Runtime Console. Each of the buttons broadcast their respective events when clicked. When the application is used with a subject, the Runtime Console should be used to control the application.

The event control buttons in the Global Events section provide the controls to create, modify, test and delete the events.

	The 'Create new event' button is used to add a new event to the Global Events list.
	The 'Rename selected event' button is used to rename an event.
	The 'Delete selected event' button is used to delete an event.
	The 'Broadcast selected event' button is used to test an event.
	The 'Delete all unused events' button is use to delete those events that are no longer mapped to certain module actions.

3.2.5 Connection Editor (purple)

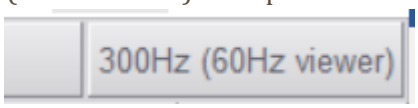
When you click on a wire (connector) in the Data Flow Editor the Connection Editor opens on the bottom of the D-Flow window. Inside the Connection Editor, the wires between each of the possible inputs and outputs are shown. All outputs are by default on the left side of the Connection Editor and the inputs are on the right side. The connection is made from the output pin of a module to the input pin of the other module.

3.2.6 Module Properties pane (orange)

The Module Properties pane can be used to easily check or adjust the properties, inputs and outputs of a selected module. After selecting a module, the right pane is automatically shown on the right side of the Data Flow Editor. The Module Properties pane automatically disappears when anything other in the Data Flow Editor than a module is selected. However, when the properties pane has been pinned, the pane is always shown.

3.2.7 Menu bar

The menu bar is located at the top of the screen. The menu bar provides access to functions in five different categories. For more information about the contents of the menu bar please refer to the D-Flow manual. On the right side of the menu bar the current frame rate is available. This part is formatted as follows: [D-flow frame rate in Hz (viewer frame rate in Hz)]. For example: 300 Hz (60 Hz viewer). This part of the screen can be used to monitor the performance of the application.



3.2.8 Status bar

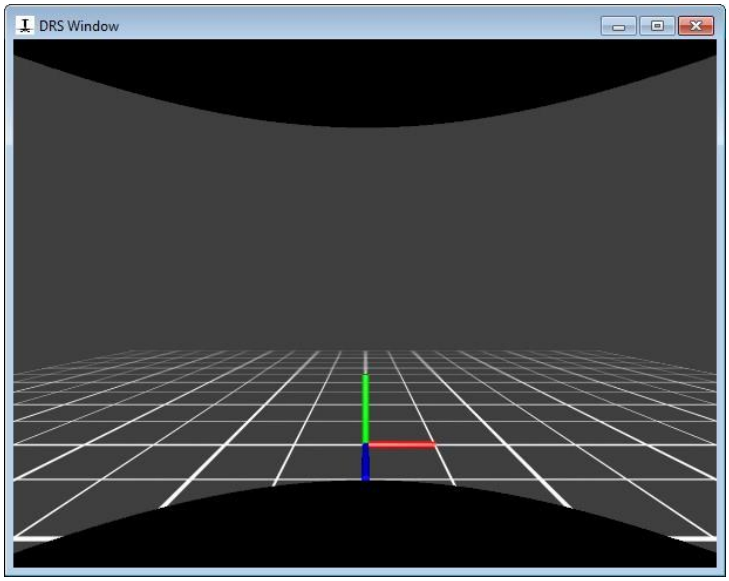
The status bar is located at the bottom of the D-Flow editor window. This section shows the time of the last status message, the last status message and the current time.

3.2.9 DRS Window

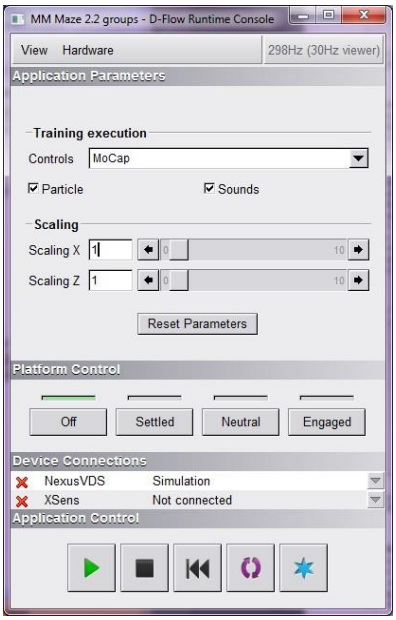
The DRS (Distributed Rendering System) window allows the operator to see what the subject sees. The operator can perform camera orbit, zoom, panning, and object selection. For virtual reality systems containing a single display, the operator DRS window can be directly cloned to be used as 3D display for the subject, omitting the use of image generator(s).

You can use the mouse to navigate through the DRS window:

- Translate the view by holding the right-button.
- Rotate the view by holding the left-button.
- Zoom in/ out by scrolling the mouse wheel.



3.2.10 Runtime Console



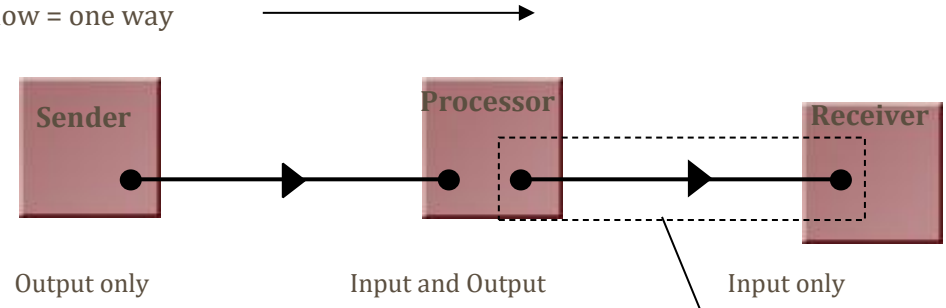
The Runtime Console is a customizable interface that can be used to control D-Flow applications. The goal of the interface is to provide a clean and efficient interface, suitable for day-to-day use of D-Flow applications. The Runtime Console GUI can be edited interactively. It is incorporated into the D-Flow editor using a module that automatically generates an output channel for each GUI element. GUI elements that link to data channels include sliders, drop-down boxes and check boxes, while buttons can be created and set to broadcast specific global events. In addition to user-defined controls, the Runtime Console contains an interface for controlling the state of the motion base, as well as a control for all other hardware devices in the virtual reality system.

3.3 Module based programming

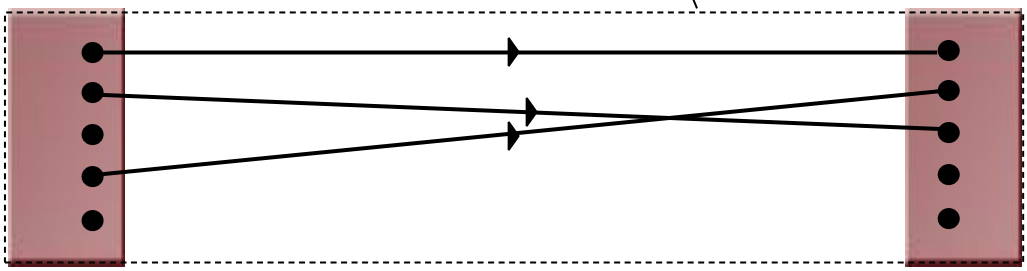
D-Flow is based on the concept of modules: manageable components with a specific functionality, which can be combined to create complex, interactive virtual reality applications. There are various types of modules present in the D-Flow software system. Some modules directly control specific hardware devices, such as a treadmill or a motion base. Other hardware modules provide access to real-time data streams from live input devices. To allow interaction between the subject and the virtual environment, there are modules that manipulate virtual objects, control playback of animations in the virtual environment, or detect collisions between objects. Finally, several modules perform specific low-level tasks, and can be regarded as building blocks for high-level functionalities. Examples are modules for controlling decision logic, variable storage and function generation. D-Flow also contains a general-purpose scripting module and a module for expression parsing. A detailed description of all modules can be found in the D-Flow Help menu.

3.3.1 Data flow

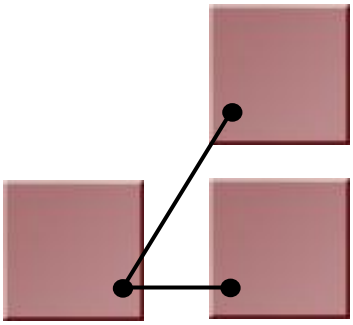
Data flow = one way



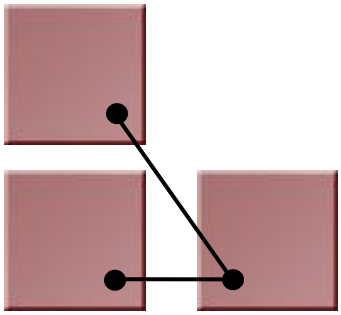
Double-click on connector to open the Connection Editor



One to many:



Many to one:

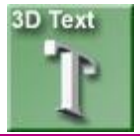


3.4 Events

In addition to data-based communication, the D-Flow framework allows for event-based communication between modules. The operator can define a set of global events, which can be broadcast by modules at specific occurrences (i.e. when a collision between virtual objects has occurred, or a value reaches a certain threshold). Next to that, each module exposes a set of module actions that affect the behavior of the module in a specific way (i.e. enabling or disabling it, showing or hiding an object, or increasing a counter). For each module instance, any global event can be set to trigger any module action, enabling maximum flexibility in event-based communication.

4 Tutorials

4.1 D-Flow Tutorial #1: Creating Data Flow

D-Flow Tutorial #1: Creating Data Flow		
Time estimation: 30-40 minutes		
Goal	The goal of this tutorial to learn how to create a data-flow in the D-Flow software. How to create values, how to perform simple data manipulations, how to display data on screen and how to save your application.	
Topics covered in this tutorial:	• Creating modules • Creating module connections • Editing module connections • Changing module settings (3D Text) • Saving an application	
Modules covered in this tutorial:		**'Generator'** This module is able to generate data based on mathematical functions depending on time (e.g. sine functions). Easy calculations like additions, multiplication, conditionals and exponentials can also be executed.
		**'Graph3D'** This module gives a live representation of data in graph-form in the 3D-scenery. For example, it could display the data created by the Generator.
		**'Expression'** This module can perform specific calculations or expressions on the data it receives and can output it through up to 6 output channels.
		**'3D Text'** This module is able to show text (in 3D) and numerals in the 3D-scenery.
References:	D-Flow 3 Help, module references: Generator module, Graph3D module, Expression module, 3D Text module.	

Introduction

With the D-Flow software, it is possible to create and obtain data from various live (or non-live) input sources. In this tutorial we are going to show you how you can manage simple dataflow from the beginning to the end.

First, what is data flow?
Data flow is the process in which data is fed trough certain modules and is used or manipulated along the way.

What are modules?
Modules in D-Flow are tools which are displayed as intuitive boxes in which you can input data, manipulate it and output it again. Combinations of modules lead to complex manipulations of data and can provide you with countless possibilities to control your hardware, virtual scenery and data capture.

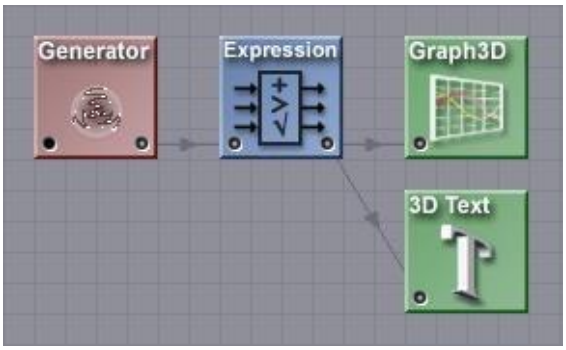
The two main D-Flow windows
When D-Flow is started, two windows will appear:
- The Editor
- The DRS Window

In the **'Editor'**, containing the Scene Explorer, Global Events section, Data Flow Editor and Module section, you can open and create applications, manipulate your scene, and record data.

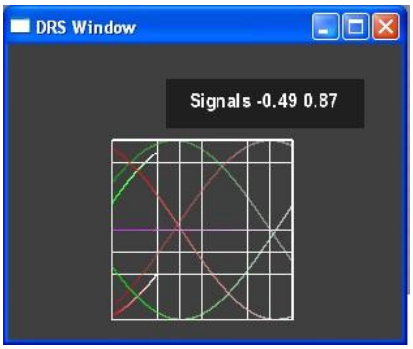
In this tutorial, we will use the Data Flow Editor, in which all your calculations/manipulations are done and your data-flow is created. In the Module section all modules are located and can be dragged into the Data Flow Editor in order to use them.

The **'DRS Window'** is the other main window, which represents the virtual 3D world you have created.

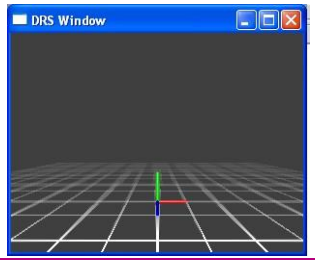
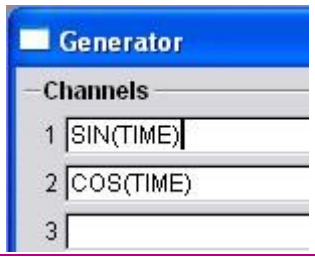
Preview of the result after completing this tutorial

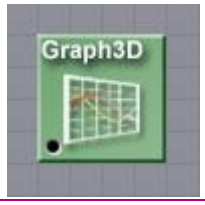
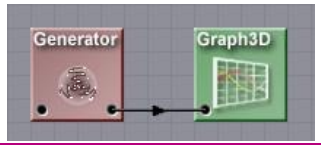
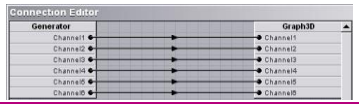


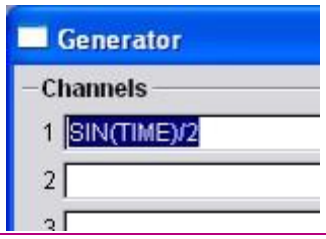
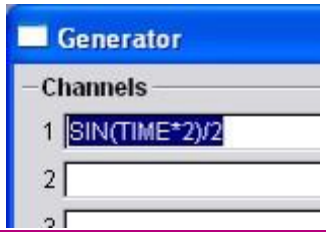
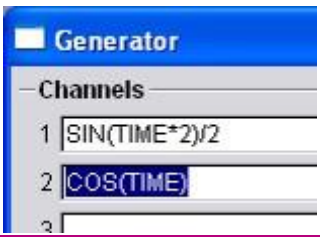
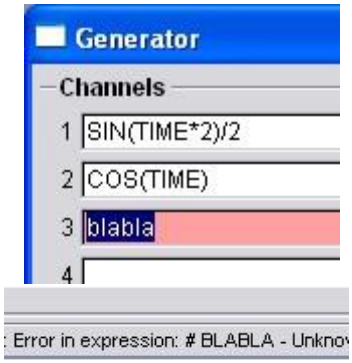
Data Flow Editor

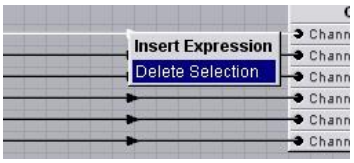
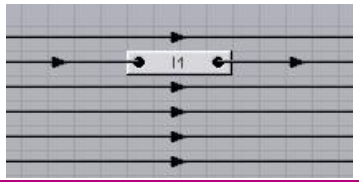
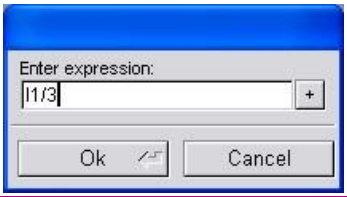


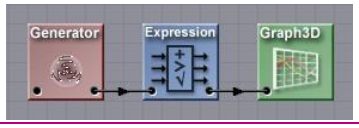
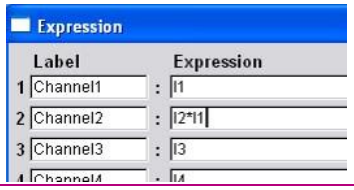
DRS Window

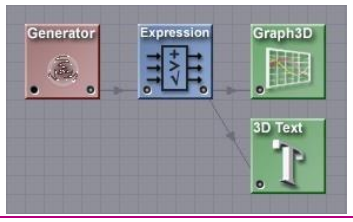
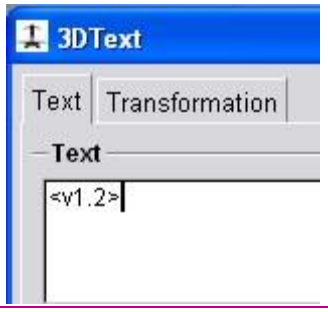
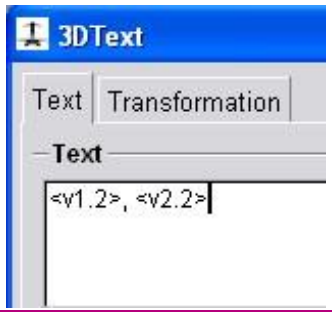
Procedure	Explanation	Illustration
1. Start D-Flow by double clicking the icon on the desktop.	Notice the two different windows. Notice that the Data Flow Editor is empty, because no application is created or loaded yet. Notice that the DRS Window shows a grid (representing the horizontal plane) and three colored axes, representing the global Y-axis (green), X-axis (red) and Z-axis (blue).	
2. Click inside the DRS Window and play with the view of the camera.	The view of the virtual world can be changed using the mouse: By holding the left button while moving the mouse you can rotate the scene. By holding the right button while moving, the world translates. The scroll button lets you zoom in and out. This makes it possible to look at your scene from different angles. Press 'R' on the keyboard to reset the view.	
Start with creating some modules.		
3. Drag the Generator module from the module section (red section) into the Data Flow Editor.	Notice that the module has two black dots in each corner. The left dot is the input pin and the right dot is the output pin.	
4. Double-click the Generator module.	Double-clicking a module will open its graphical user interface. Now you see the 6 different channels in which the generator can perform calculations and create data. The default settings from the Generator show 'SIN(TIME)' in channel 1 and 'COS(TIME)' in channel 2. This means that the sine function of the variable 'TIME' is calculated and outputted in channel 1, and the cosine of the variable 'TIME' is calculated and outputted in channel 2.	
5. Delete COS(TIME) in channel 2. Press Enter to confirm the change.	We only want to use the output of channel 1 for now. Notice that the 'COS(TIME)' text box turns blue when deleting this text. In D-Flow is it necessary to confirm every adjustment/ change by hitting the enter button.	

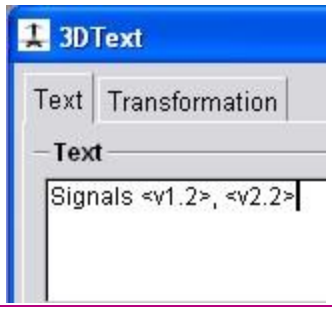
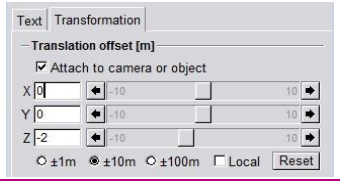
Procedure	Explanation	Illustration
6. Drag a Graph3D module (from the green section) into the Data Flow Editor.	Notice a grid appears in your DRS Window, this is the outline of the graph we are going to create. Also notice that the module only has an input pin, and no output pin. This means the module is only able to receive data, not output data.	
7. In the DRS Window: zoom out using the scroll button. Translate the view by holding the right button and moving the mouse. Rotate the view by holding the left button and moving the mouse.	This way you can change the view of the graph so you can see its full size. Play with the view until you think you can see the entire graph.	
8. In the DRS Window: Press 'a' and 'g'	Pressing 'a' will hide the axis. Pressing 'g' will hide the grid. Press them again to make them visible again. In order to use shortcut keys for the DRS Window, the DRS Window needs to be active. To activate the window, just click in it once and you are ready to use the shortcut keys. The grid and axes are useful as a reference when you are creating a scene. But in most cases you will hide them once your scene is coming together.	
Now we are going to create the data flow.		
9. Click on the output pin of the Generator, hold and drag a line to the input pin of the Graph3D.	Notice you have created a line linking the two modules.	
10. Click on the line (Connector) between the Generator and Graph3D module. When selected it will appear in white and bold.	Now you have opened the Connection Editor. On the left side you see the output channels from the Generator, and on the right side you see the input channels of the Graph3D. The black lines are all connections between the Generator and the Graph3D.	
11. Double-click on the Graph 3D module.	Double-click on a module will open its graphical user interface. Notice that the Graph 3D module has a lot more settings than the Generator module.	

Procedure	Explanation	Illustration
12. Click the 'Play'-button in the Generator's user interface. And click the 'Play' button in the Graph3D user interface.	By clicking the 'Play' button the Generator will start creating data (in this case the sine function of the time). The data goes through the connection line, into the Graph3D. When the Graph3D starts playing, the data is displayed in the DRS Window. Notice that all modules have their own play and stop controls. This allows you to control modules separately.	
We have seen the basics of creating data flow. Now we are going to take it a step further.		
13. Open the Generators user interface in case you closed it.	We are now going to change the output signal of the Generator.	
14. In Channel 1 type: SIN(TIME)/2 and press 'Enter'	This means that the sine function of the time is now divided by 2. Notice that the amplitude of the signal in the Graph3D twice as small. It is a good habit to always press 'Enter' after you have typed in a new expression. This is to confirm your input.	
15. Now type: SIN(TIME*2)/2 and press 'Enter'.	This means that the TIME is multiplied by 2 before the sine function is calculated. Notice that the frequency of the sine in the DRS Window is twice as fast. This shows that you can perform calculations on the variables before they become Generator output.	
16. Type in Channel 2 of the Generator module: 'COS(TIME)', and press 'Enter'.	Notice how an extra (green) signal has appeared in the Graph. This is the cosine of the TIME. In case D-Flow does not recognize your input, the expression field turns red.	
17. Type a meaningless expression (for example (blabla) in Channel 3.	Notice that the field becomes red. Look at the log statement in the left lower corner of the D-Flow editor window. This tells you why the expression is not correct.	

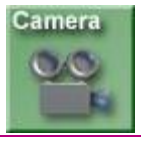
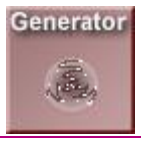
Procedure	Explanation	Illustration
18. In the Connection Editor: Right-click on the Channel 1 – Channel 1 – connection. Press 'delete selection'. Press 'Yes' in the warning-window. If the Connection Editor is not yet open, simply click on the connection between the generator and the Graph3D module to open it.	This deletes the connection between the first channels. Notice how the red (the first) signal has disappeared from the Graph.	
19. In the Connection Editor: recreate the connection by dragging from the black (output) pin next to Generators' Channel 1 to the input pin of the Graphs Channel 1.	Notice the red signal reappears.	
It is also possible to do simple calculations in the connection lines.		
20. Double-click on the line between Channel 2 on the Generator and Channel 2 on the Graph3D module.	Notice the gray box that appeared in between the line. This box is called a math box. By default it feeds through the data it receives on input 1 (I1). But the math box gives you the possibility to manipulate this input.	
21. Double-click the math box.	This is the expression window of the connection.	
22. Type: 'I1 / 3' and press 'Enter'. **NOTE:** Make sure you type I1 instead of L1.	I1 (or i1) means Input 1. Notice the amplitude of the green (Channel 2) – signal is divided by three.	
Now we are going to take a bigger step. In order to do more difficult calculation than *3 or /3 a special 'Expression' module is used. We are going to feed the data through the Expression module instead of making a direct connection between the Generator and the Graph.		

Procedure	Explanation	Illustration
23. Drag the Expression module from the module section (blue-section) into the Data Flow Editor.	This module is almost similar to the Generator module but does not contain the TIME-function and is used to perform calculations between input and output modules, instead of generating data. Notice it also has an input and output pin.	
24. Delete the connection between the Generator and the Graph (right-click and choose 'delete selection').	You can also use 'Delete' on your keyboard.	
25. Make a connection between the Generator's output and the Expression's input and another between the Expressions-output and the Graph3D input. Make sure all modules are playing (if not press the 'Play' buttons).	Notice that the signals appeared again in the DRS Window. The data from the Generator is now fed through the Expression to the Graph.	
26. Double-click on the Expression module in the editor.	Notice the numbers behind the '=' on the right side. These are the output-values of the Expression module that are outputted to the Graph3D module. I1 stands for the input received in the Expressions first input channel (Input1). I2 stands for the input received in the second input channel (Input2), and so on. You are free to use these input values (I1, I2, I3...) in other channels than where they are by default.	
27. Type in Channel 2: 'I2 * I1' and press 'Enter'.	Now the output of Channel 2 is equal to the input of Channel 1 times the input of Channel 2. Notice the green signal has changed.	
Now you know how to create data, perform calculations and how to display the values in a graph. Now let's look at how to display them as numbers.		

Procedure	Explanation	Illustration
28. Drag a 3D text module from the module section (green section) into the Data Flow Editor.	The 3D Text module is a module that allows you to display numbers and text in your DRS Window. Notice that the module only has an input pin but no output pin.	
29. Make a connection between the Expression and the 3D Text module. (Keep the existing connections.)	Modules can have multiple connections. Notice that nothing appeared on the DRS Window yet. This is because we have to specify the 3D Text 'output' first.	
30. Double-click on the 3D Text module.	This is the graphical user interface of the 3D Text in which you can specify which output it will display.	
31. In the white-area type: '<v1>'	This will display the value of input 1. Notice that the number 0 or -0 appears in the DRS Window.	
32. In order to give the number more decimals, type in the white area: '<v1.2>'	This will display the value of input 1 with 2 decimals. For three decimals type <v1.3> etcetera (up to 6 decimals).	
33. Now type: '<v1.2>, <v2.2>' in the white area of the 3D Text module. To display the values on separate lines simply hit enter after the comma.	Notice that the billboard can also display multiple numbers (with a maximum of 9).	

Procedure	Explanation	Illustration
34. Now type: 'Signals' in front of '<v1.2>, <v2.2>'.	3D Text modules are able to display text and numbers at the same time. This way you can specify the number you display in order to make it understandable for the viewers.	
35. Click on the 'Transformation' tab of the 3D Text module.	Here you can see the X, Y and Z positions of the text with respect to the camera.	
36. Click on the slider next to the X-position and drag it to the right.	Notice that the position of the text in the DRS Window changes. The same technique goes for the Y and Z position. Changing the X translation moves the text left and right, Y moves the text up and down and z moves it towards or away from the camera.	
37. Change the position of the text until you like it.		
Now we have created data, performed calculations and displayed them. Now let's save the application so we can load it in the future. D-Flow has its own directory in which Applications, Data, Scenes and Sounds are saved in order to keep them organized and clear		
38. In order to save the application click: File | Save As…	This automatically opens the save-window and opens the Applications folder in the CAREN Resources directory. This is the folder in which all applications should be saved.	
39. Create a folder called 'Training apps'.	Here, we will store all applications that are created during this training.	
40. Now specify the name 'My first application' and click 'Save'.	The file is now saved as a .caren file in the CAREN Resources directory and we can open it whenever we want. Notice the application name is the left upper corner of the D-Flow Editor Window.	

4.2 D-Flow Tutorial #2: Getting started with a scene

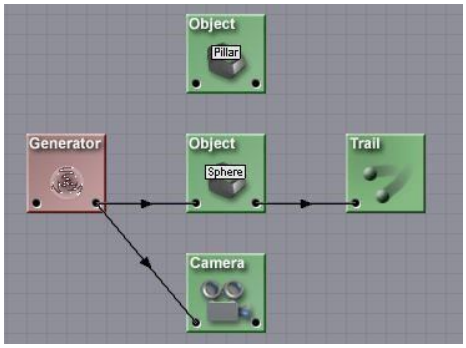
D-Flow Tutorial #2: Getting started with a scene		
Time estimation: 30-40 minutes		
Goal	The goal of this tutorial is learning how to create a scene, create and move objects and how to move around in the scene.	
Topics covered in this tutorial:	Starting a new application User camera and scene camera Moving the scene camera Adding and manipulating scene objects Creating a trail behind a moving object	
Modules covered in this tutorial:		**'Camera'** This module controls the position and orientation of the user's point of view. It can be set to a specific point, set to specific objects or moved individually. It is possible to move the camera through the scene, which simulates the observer moving through the scene.
		**'Object'** This module controls the transformation and animation of a single object. It is possible to change the object's position, its scaling, its color and use it to collide with other objects. This way objects can be used as targets, distracters or to create a scenery.
		**'Trail'** A trail shows a colored path from its new position to its old positions. It shows a trail behind a moving object.
		**'Generator'** This module is able to generate data based on mathematical functions depending on time (e.g. sine functions). Easy calculations like additions, multiplication, conditionals, and exponentials can also be executed.
References:	D-Flow 3 Help, module references: Camera, Object, Trail and Generator module.	

Introduction

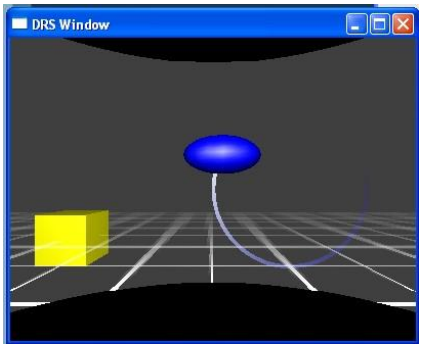
In D-Flow it is possible to create simple environments filled with objects, manipulate the objects and move through the scene. This way objects can be used in games as targets, distracters or to create depth in the scene. By doing so the game becomes more interesting for people to watch and play.

In the previous tutorial we learned a little bit about how data-flow works, in this tutorial we are going to learn how to use this data-flow to move objects.
We are also going to use the Camera module, which allows the viewer to track objects or move along a path. By using a Camera module it is possible to specify how the observer is viewing the virtual world.

Preview of the result after completing this tutorial

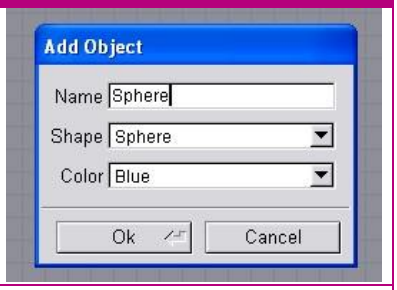
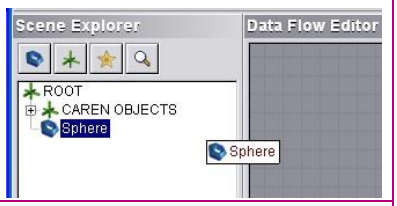


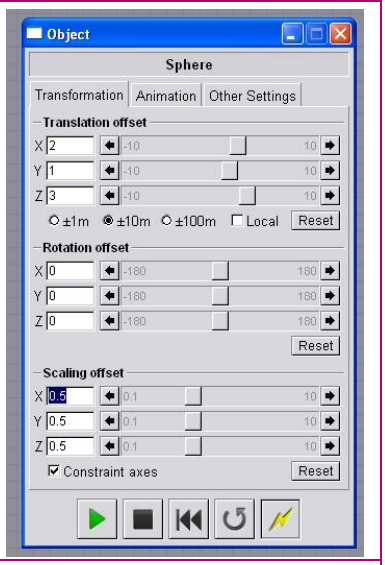
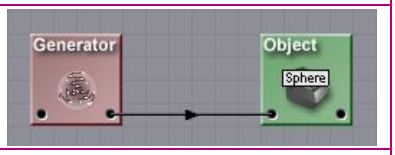
Data Flow Editor

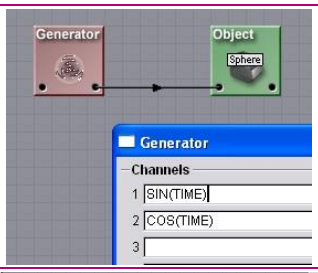
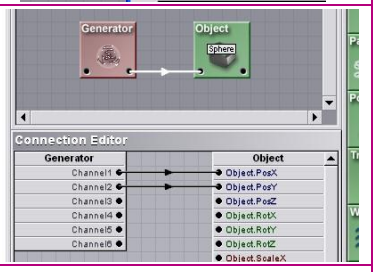
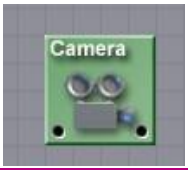
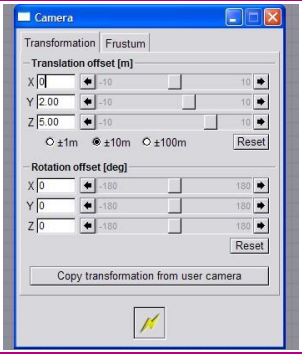


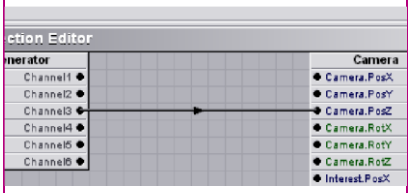
DRS Window

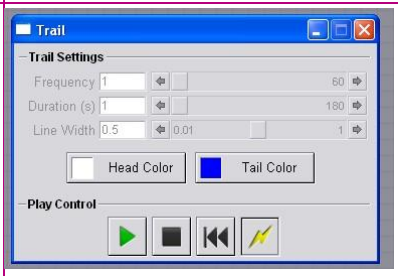
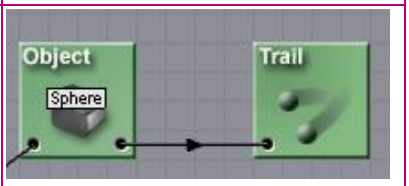
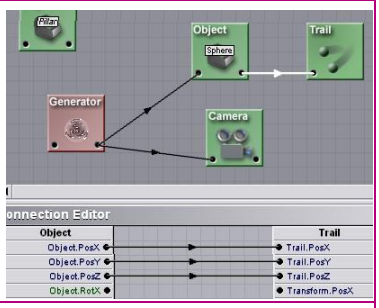
Procedure	Explanation	Illustration
1. Go to Menu Bar | File | New Application.	D-Flow will start with an empty application, just like you would start a new 'MS Word' file. Similar to files created in MS Word, you can also alter the application and save it with a different name in order to create multiple versions.	
2. Click 'Scene' and choose 'Add Object'.	This opens a window, which lets you add a simple object like cones, cubes, cylinders and spheres to you scene. Settings like name, shape and color can be set here.	
3. Type in the Name-field: 'Sphere'.	This changes the name of the object from 'user_object' to 'Sphere'. An object can be given any name, which makes them easier to distinguish.	

Procedure	Explanation	Illustration
4. Change Shape from 'Cone' to 'Sphere'. Change the Color from 'Red' to 'Blue'.	This changes the shape and color of the object you are going to put in the scene.	
5. Click 'Ok'.	Notice that in the DRS Window an object has appeared. You can click and rotate inside the DRS Window to look around the object. Also, notice that in the 'Sphere' object has been added to the scene root in the Scene Explorer section.	
6. Drag the 'Sphere' from the Scene Explorer into the Data Flow Editor.	D-Flow automatically creates an object module when a scene object is dragged into the Data Flow Editor. You could also create an object module first and then drag the scene object on top of the object, but the first method is quicker. Notice that the object module has an input and output pin. Also, notice that the sphere is surrounded by a white wired box. This box shows you that you have selected this particular object in the scene explorer.	
7. Double-click on the Object module.	This opens the Object module's user interface. It contains three tabs in which you can manually change the object's position, rotation and scale offset and animation and material settings.	
8. Click the 'Lightning button in the objects user interface.	This will hide the object in the DRS Window. The lightning button is used in various modules like 3D Text, 3D Graphs and the Camera. This button is used to toggle them on or off in the scenery without removing them from the Data Flow Editor.	
9. Click the 'Lightning' button again. **Note:** Make sure that the object is visible before continuing to the next step, e.g. the lighting button is yellow.	The object will reappear.	

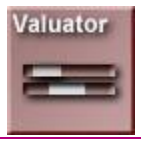
Procedure	Explanation	Illustration
10. Select the 'Transformation' tab and change the Translation offset of the X to 2.	This moves the object from the origin to X = 2, changing X moves the object along the X-axis (the red axis). The transformation can be changed using the slider, typing in the white field next to the X or using the modules input (we will get to that later).	
11. Change the offset of the Y to 1 and the Z to -3.	This moves the object along the Y (green) and Z (blue) axis. As before, don't forget to press Enter after changing the offsets.	
12. Set scaling offset of the X-slider to 0.5	Notice that the object has changed in size. Also notice that the Y and Z sliders also changed to 0.5. This is because the constraint axes box is checked. Uncheck the box lets you change all axes separately.	
13. Uncheck the constrained axes box and change the X-slider to 1	Now you have created a blue elliptic figure.	
Let's start moving the object. In order to move an object, we need to generate data that is used for its position input along time.		
14. Drag a Generator module from the Module section (red section) to the Data Flow Editor	We will use the Generator to create a value that changes over time, and we will link this to the Object module.	
15. Make a connection between the Generators output pin and the Objects input pin.		
16. Click on the connection line between the Generator and Object module.	Notice that no connections were made automatically. This is because D-Flow does not recognize a clear match between the output and input channels. This does not mean that connections are not possible.	

Procedure	Explanation	Illustration
	On the left you will see the various output channels of the Generator. On the right you will see all possible input channels of the object: position, rotation, scaling, color, show.bool (makes it disappear or appear) and the animation speed.	
17. In the Connection Editor: Create a connection between the Generator's Channel 1 and the object's Object.PosX.	This way the output of Channel 1 is used as the position input for the object's X-position.	
18. Open the Generator's user interface (double-click) and click on the 'Play' button.	Now you can see that the object starts to move according to its input 'SIN(TIME)'. Notice that the object moves one meter to right and 1 meter to the left. (one section of the grid is equal to one meter)	
19. In the Connection Editor: Create a connection between the Generator's Channel 2 and the object's Object.PosY.	The object now is moving in a circle.	
Now let's add a camera.		
20. Drag a Camera module from the module section (green section) into the Data Flow Editor.	Notice that you cannot move the view in the DRS with the mouse anymore. The position of the camera is now controlled with the Camera module. This module has similar settings regarding position and rotation compared to the Object module.	
21. Open the Camera's user interface (double-click on the module)	Here you can see the translation and rotation offsets the camera can be given.	
22. Change the Translation offset of the X and Y both to 1.	The camera has now moved to the right and down.	

Procedure	Explanation	Illustration
23. Try the 'Lightning' button in the Camera's user interface. **Note:** Make sure you toggle on the camera again before you proceed.	This button will toggle off the 'scene camera' and enables the 'user camera' again. The DRS Window mouse functions work again once the scene camera is toggled off. **Note:** the mouse functions for translating and rotating the DRS window will not work when the camera is on.	
Let's create another object and then start moving the camera.		
24. Click 'Scene' > 'Add Object' Change the name to 'Pillar', the Shape to 'Cube', the Color to 'Yellow' and click OK	Now you have created a yellow cube which is located in the middle of your scene (the origin).	
25. Drag the object 'Pillar' from the scene explorer section to the Data Flow Editor.	A new Object module appears. Every object has its own module so you can manipulate them individually.	
26. Double-click the 'Pillar' module. Change the Translation Offset X to '2' and the Z to '-5'	This will move the object.	
Now let's start moving the camera.		
27. Open the user interface of the Generator. Type in Channel 3: 'TIME*0.1' and press 'Enter'.	Now we have created a signal that increases over time (+ 0.1 for every second). With this signal, we are going to control the camera motion.	
28. Create a connection between the Generator module and the Camera module.		
29. Open the Connection Editor for this connection and create a connection between Channel 3 of the Generator and the Camera PosZ.	Now the camera will move in the Z direction according to the output of Channel 3 of the Generator. Notice that in your DRS Window the objects are far away, almost out of sight. This is because TIME has been increasing since you have clicked 'play' in the Generator module's user interface.	
30. Click the reset-button inside the Generator module's user interface.  = Reset button	This will reset the 'TIME' to '0'. Because the Generator is still playing, 'TIME' immediately starts to increase again, which means that the camera immediately starts to move in the Z-direction. In other words, the camera is slowly moving backwards.	

Procedure	Explanation	Illustration
This is a simple example of how to control the camera position. Camera positions can also be controlled via more difficult calculations, an object, a joystick and marker input etcetera. Now we are going to add a trail to the moving object. A trail draws a colored path behind the object on the positions the object has previously been.		
31. Press the 'Stop' and 'Reset' button in the Generator's user interface.		
32. Drag a Trail module from the module-section (green section) into the Data Flow Editor.	Notice the module only has an input pin.	
33. Double-click on the Trail module.	In this interface, you can change the frequency, duration, line width and colors of the trail.	
34. Make a connection between the 'Sphere' Object module and the Trail module.		
35. Select the connection between these modules.	Notice that in the connection editor a lot of connections have been made by default.	
36. Delete all connections except for the ones between the Objects PosX, PosY, PosZ and the trails PosX, PosY and PosZ.	Now the position of the object and the trail are linked together.	
37. Click the 'play' button in the Generator and Trail module's user interface	Notice that the object starts moving again and a white-to-blue trail appears behind it.	
38. Now let's save the application. Click 'File | Save Application as'. Save the application as 'MovingSphere' in the folder 'Training apps' you have created earlier.		

4.3 D-Flow Tutorial #3: Getting started with events

D-Flow Tutorial #3: Getting started with a events		
Time estimation: 30-40 minutes		
Goal	The goal of this tutorial is to learn how the event system in the D-Flow software works. This is done by creating a small application.	
Topics covered in this tutorial:	• Triggering events based on a specified condition • Triggering events based on collisions between objects • Using the Runtime Console • Using the Module Properties pane • Counting events	
Modules covered in this tutorial:		**'Event'** This module broadcasts an 'event' to all modules when the state of a specified condition changes from true to false or vice versa.
		**'Stopwatch'** This module displays a time counter on a stopwatch and is able to output the time as a value.
		**'Collision'** This module detects when two (or more) objects collide with one another and broadcasts the occurrence of the collision as an event.
		**'Valuator'** This module has a number of sliders with a predetermined range which can be changed during runtime. This module is helpful to simulate output in the testing phase of application creation.
		**'Counter'** This module counts the amount of times an event has occurred and output this value.
References:	D-Flow 3 Help, Module References: Event, Stopwatch, Collision, Valuator and Counter module.	

Introduction

The event system in the D-Flow software is a very important part of each application that is build. With the event system you are able to control (change) the behavior of the modules (module actions), in the application at the time specific events (global events) occur. Without this system you would have to activate all the different module actions of the different modules by yourself. This is impossible, as you do want some modules to start at the exact same time and others only in reaction to a certain event. You can imagine that you would never be able to this. This is why D-Flow has an event system.

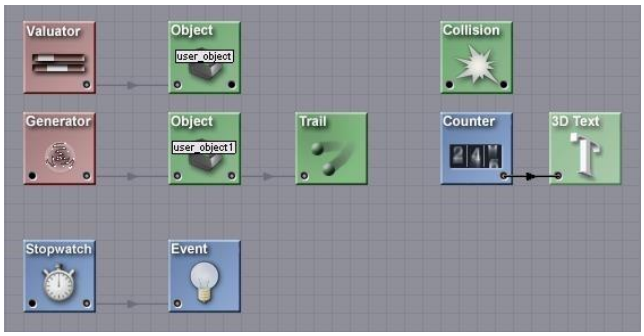
One of the modules used in this tutorial is the Event module. As soon as a certain condition is met (a certain value is reached), this module broadcasts an event (message) to all modules. This message 'tells' all the modules that the specific event has occurred. The modules that have been programmed to react to that specific event will use this signal to perform their specific module action.

For example:
A Stopwatch module starts counting down from '5' and reaches '0' seconds. An Event module is set up to detect this specific condition and is programmed to broadcast the message 'Go' to all modules. A 3D Text module with the text 'Go' is programmed to hide its text unless he gets the message 'Go'.
While the Stopwatch has not reached '0', the text of the 3D Text module is hidden. When the Stopwatch has reached '0', the text 'Go' appears on screen.

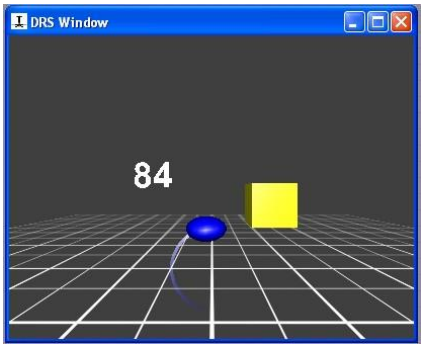
An important thing to know is that when an event is broadcast, it is sent to all modules, however only those modules that are programmed to react to the message will come in action. For this 'message service', no connections between modules are needed. The event system works 'wireless'.

In this tutorial we will explore the possibilities of the Event module and Collision module. The first is able to broadcast events based on specified conditions, the second based on the collision of two scene objects. In order to save time on setting up a new application, we will proceed with the application you have created in Tutorial #2.

Preview of the result after completing this tutorial

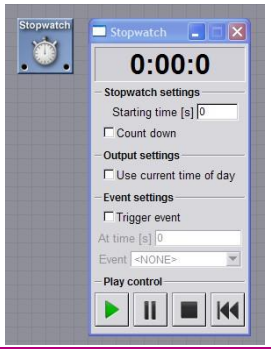


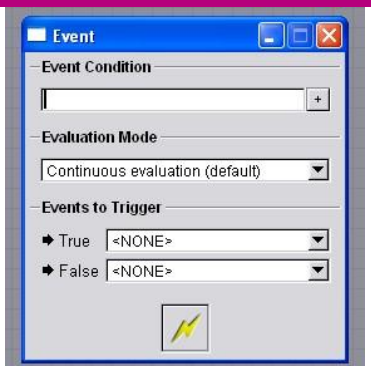
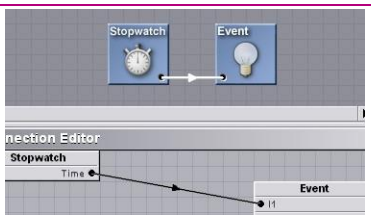
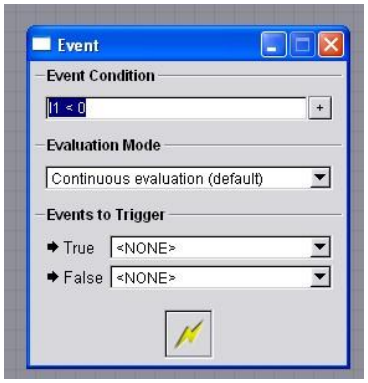
Data Flow Editor

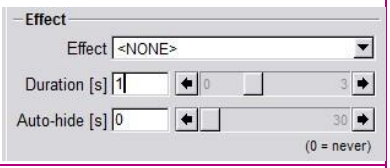
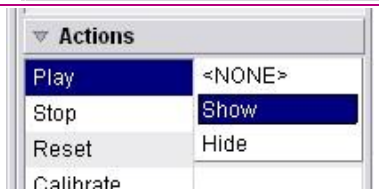
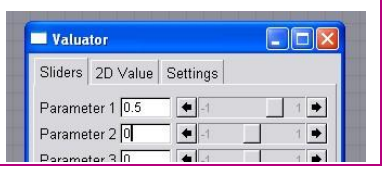


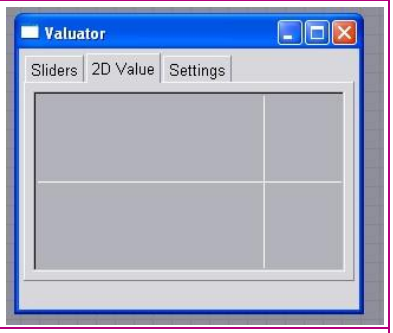
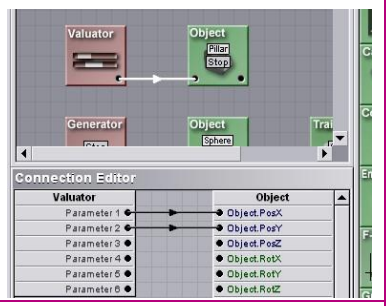
DRS Window

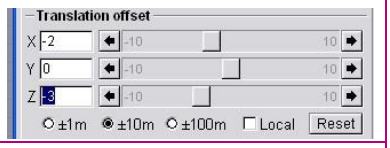
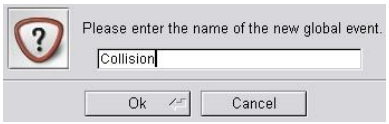
Procedure	Explanation	Illustration
1. Click 'File | Open Application'		
2. Go to the folder 'Training apps' (or the name you chose in Tutorial #2) and open the application you have created in Tutorial #2.		
3. In this case we do not want the camera to move so: Delete the Camera module.	Now you can adjust the point of view in the DRS Window manually again.	
On the left you see the Global Events window containing all events which have been created by default (Play, Stop, Reset, Calibrate and Action). When you double-click on an event in the Global Events window, the event will be broadcast to all modules. This also goes for all events we will create in the future.		
4. Open the Runtime Console by pressing F2. **Note:** this works only if the mouse is in the D-Flow editor window and not in the DRS window.	The Runtime Console is used to control the application in runtime. The console contains the play control buttons 'Play', 'Stop', 'Reset'. 'Calibrate' and 'Action. In case you have created runtime parameters for your application, than this panel also contains of parameters that are used to alter module settings during runtime.	
5. Click the 'Play' button.	Pressing this button will the broadcast of the 'Play' event. All modules, by default, are programmed to start Playing when this event is broadcast.	
6. Click on the Generator module. The 'Module Properties' pane appears on the right side of your screen	In the 'Actions' section you can determine to which events the module reacts, and what the reaction should be. All modules have their own event mapping. Notice that the module actions also can be adjusted by right-clicking on a module and select 'Edit Event Mapping'.	
7. Set the Module Action next to the Global Event 'Action' from <NONE> to 'Pause'	Now the Generator's reaction 'Pause' is linked to the Global Event 'Action'. This means that when 'Action' occurs the Generator will pause until the event 'Play' occurs again.	

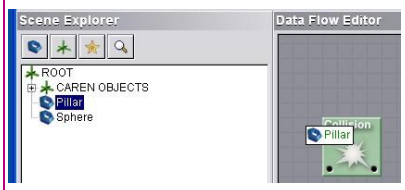
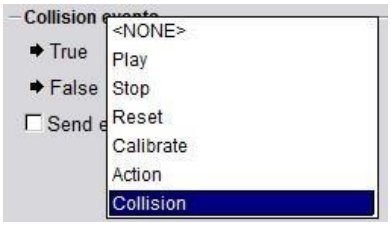
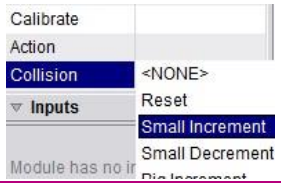
Procedure	Explanation	Illustration
8. Click the 'Action' button in the Runtime Console.	The Generator pauses (notice the blue sphere has stopped moving).	
9. Click the 'Play' button in the Runtime Console.	This triggers the reaction 'Play' in the Generator, which makes the output start to run again, so the Object starts moving again.	
10. Click the 'Stop' button in the Runtime Console.	This stops all modules.	
Now we are going to create a stopwatch which will count down before all modules start Playing and a text which will announce the start.		
11. Drag a Stopwatch from the module section (blue section) into the Data Flow Editor.		
12. Double-click on the Stopwatch module.	This opens its user interface in which you can see its time and where you are able to adjust the settings. In this case we want the time to count down from 5.	
13. Set starting time to 5 and press enter to confirm your input. Tick the 'Count down' box.	The time in the timer has gone to 5 and the Stopwatch will count down when playing.	
14. Click the 'Play' button in the Stopwatch user interface (not the one in the Runtime Console).	Notice the time starts running down and continues after reaching '0' as a negative number.	
15. Click the 'Stop' button and click the 'Reset' button in the Stopwatch user interface	Notice that with the Stop button it stops, and with the Reset button it goes back to 5 (our initial value).	
16. Drag an Event module from the module section (blue section) into the Data Flow Editor.	This creates the Event module.	

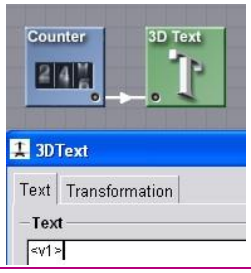
Procedure	Explanation	Illustration
17. Double-click on the Event module.	Now you will see the user interface of the module, where you can specify the event condition. Besides that, it shows the Evaluation mode (when does the module check if the condition is met or not) and the Events to trigger (which event is broadcast when the condition is met or not).	
18. Create a connection between the Stopwatch and the Event module and click the connection line	In the Connection Editor, you will notice that the 'Time' output is automatically linked to the first input channel (I1) of the Event.	
19. In the Event module's user interface type in the Event Condition: ' I1 < 0' **Note**: Use an I and not a L, because of Input 1. Don't forget to press Enter.	This means that when input 1 (Time output of the Stopwatch) is lower than '0', the Event Condition is true.	
20. Under Events to Trigger set to event behind 'True' from <NONE> to 'Play'	This means that when condition is true (I1 < 0) then the event 'Play' is broadcast to all modules. Just like when you click the 'Play' button in the Runtime Console.	
21. Click the 'Stop' and 'Reset' buttons in the Runtime Console.'	This stops the application and resets the stopwatch. So when 'play' is pressed again, the stopwatch starts running and the entire process is starting over again.	

Procedure	Explanation	Illustration
Now we want to produce a billboard saying start when the Stopwatch reaches 0 and the application starts playing.		
22. Drag a Billboard module from the module section into the Data Flow Editor and double-click the module. Type the text 'Start!'	This creates a module with the text 'Start!'	
23. Enter a '2' in the textbox for 'Auto hide after time' (on the Appearance tab).	This means that the billboard will be hidden automatically after 2 seconds from the moment it is being showed.	
24. Click on the Billboard module and go to the 'Action' section of the Module Properties pane on the Right Hand side of the D-Flow editor window. Next to the Global Event 'Play' set the Module Action to 'Show'.	This means that when the event 'Play' is broadcast by the Stopwatch and the application starts playing the text will be shown. After 2 seconds the billboard will automatically be hidden.	
25. In the Stopwatch user interface press 'Play'.	Notice that time will run down and when it reaches 0 the billboard appears for 2 seconds, and the object starts moving.	
26. Stop and reset the Stopwatch by using the buttons in the Runtime Console.		
We have modified our old application a lot by now, so we are going to save the application under a new name before we proceed.		
27. Click 'File' > 'Save Application As...'. Go to the folder 'Training Apps' and save the application with the name 'UsingEvents'.		
Now we are going to start learning how to manually control an object.		
28. Drag a Valuator module from the module section (red section) into the Data Flow Editor and open the Valuator's user interface.	Here you can see 6 channels. The Valuator can output to other modules. It works in a similar way to the Generator, only the values are changed manually in this module. In the first tab you see the current values of the different parameters (or channels).	
29. Change the slider of Parameter 1 to 0.5.	This changes the output value of Parameter (Channel) 1 from 0 to 0.5	

Procedure	Explanation	Illustration
30. Click the 2D value tab.	This tab is a tab which combines parameters 1 and 2. This way you can manually slide 2 parameters simultaneously. This lets you for example move objects in 2Dspacing (which we are going to use).	
31. Drag the cross in the gray field to the lower left corner and go back to the Sliders tab.	Notice that parameters 1 and 2 have respectively changed from 0.5 and 0 to -1 and 1.	
32. Go back to the 2D Value tab.	The horizontal axis on the field represents Parameter 1 for -1 to 1 (from left to right). The vertical axis represents Parameter 2 from -1 to 1 (from top to bottom, which is different from mathematical axis!).	
33. Go to the Settings tab.	Here you can change the range of all parameters.	
34. Change the Parameter 1 Setting from -1: 1 to -10:10 Change the Parameter 2 Setting from -1: 1 to -10:10	This changes the range of both parameters. Putting the cross in the 2D Value tab to the left will produce a value of 10 for Parameter 1, to the right will produce a value of 10. Moving the cross to the top will lead to a value of -10 in Parameter 2, to the bottom it will lead to a value of 10.	
Now we are going to connect the Parameter output of the Valuator to position the Object 'Pillar' (the yellow cube).		
35. Create a connection between the Valuator and the Object module 'Pillar' and click on the connection.	On the left you see the output parameter values of the Valuator. On the right you see the input values of the Object.	
36. In the Connection Editor: Connect Parameter 1 to the Object.PosX and connect Parameter 2 to the Object.PosY.	Now the parameter values 1 and 2 of the Valuator are linked to the objects X and Y positions. When the parameters in the Valuator change the object's position will also change.	

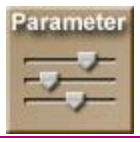
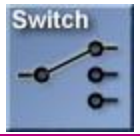
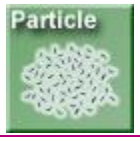
Procedure	Explanation	Illustration
37. Go to the 2D Value tab in the Valuator module and move the cross.	Notice that when the cross goes to the right, the object goes right and when the cross goes up the object goes down. This is because the output of the second channel is negative when the cross is at the upper part of the area. We will add a connection in the connection editor to flip the sign for the translations in Y-direction.	
38. In the Connection Editor, double-click on the connection between Parameter 2 and 'Object.PosY'.	An expression box will appear within the connection.	
39. Double-click on the expression box and enter '-I1' in the expression field.	-I1 means that all negative inputs become positive and vice versa.	
40. Move the cross in the 2D value tab of the Valuator again.	Notice that the object now goes up when you move the cross up. This behavior is more logical.	
The object however only moves along its X and Y axis and not along the Z axis. Therefore, if we want to collide with the blue object, they need to have equal Z values.		
41. Open the user interface of the object 'Pillar' In the Transformation tab, change the Translation offset Z from 5 to -3.	Now the Z values of both objects are equal.	
42. Go to the 2D Value tab in the Valuator module and move the cross.	Now notice that the yellow object can touch and move through the blue object.	
Now you are able to control the position of the yellow cube manually we are going to create a collision event. A collision event is an event which occurs when two object touch. This means that when the yellow cube hits the blue object an event will be broadcast. First we will create a new event and then we will create the Collision module.		
43. Under Global Events, click the 'Create new event' button (the button under the 'Play' button showing a blank sheet of paper).	This opens the window where you can create a new Global Event.	
44. In the name field type: Collision and press 'Ok'.	This creates a new Global Event which can now also be selected in all Event mappings of all modules. Notice that in the Global Events window 'Collision' is listed as well.	
45. Drag a Collision module from the module section (green section) into the Data Flow Editor.	This creates the Collision module. Here we can specify which object collisions should trigger an event the moment they collide. First we need to specify the objects that are able to collide with each other, or so called collision objects.	

Procedure	Explanation	Illustration
46. Open the Collision module user interface and place the user interface besides the Collision module icon.		
47. From the Scene Explorer window (top left), drag the 'Pillar' object on top of the Collision module in the Data Flow Editor.	The object 'Pillar' is now added to the list of Collision objects. The first added object is by default set as 'Primary Collision Object'. Collisions only count when it involves a collision between the primary object and another.	
48. From the Scene Explorer window drag the 'Sphere' object on top of the Collision module in the Data Flow Editor.	Now the object 'Sphere' is added to the list of Collision objects. This means that from now on when these two objects collide, this will be detected by the collision module, and an event can be broadcast.	
49. In the Collisions Interface under 'Collision Events' set True from <NONE> to 'Collision'.	This means that when a collision between the two object 'Pillar' and 'Sphere' is true the event 'Collision' will be broadcast. For False we do not want an event at this moment.	
Now that the collision is detected, we want to find a way to show that the collision is detected. This is where the counter comes in. A counter counts the number of times an event has occurred and outputs the value.		
50. Drag a Counter module from the module section (blue section) into the Data Flow Editor.		
51. Open the Counter module's user interface.	The top part of the window shows the current count, which is '0'. The lower part shows the initial value and the size of increments and decrements.	
52. Go to the Module Actions of the Counter Module (click on the module, go to the Actions section of the Module Properties pane).	In the Module Properties pane, notice that the Global Event 'Collision' is also in the list of Global Events.	
53. Change for the Global Event 'Collision', the Module Action from <NONE> to 'Small Increment'	Now we programmed the Counter to increase the value by one point, every time the event 'Collision' occurs.	
Now let's test if it works.		

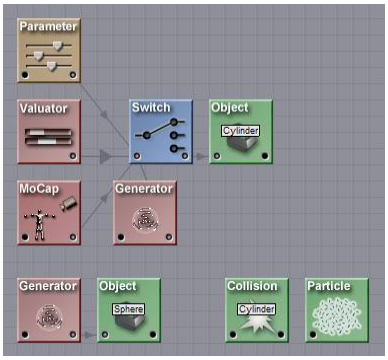
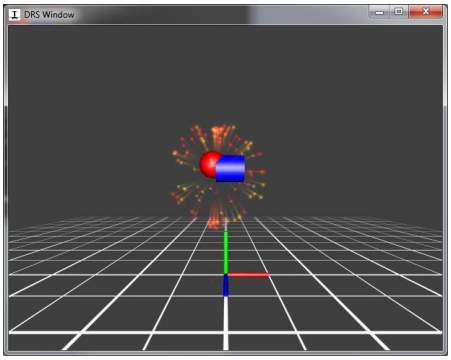
Procedure	Explanation	Illustration
54. Stop and reset the application using the buttons in the Runtime Console. **Note:** if it is not yet open, open the Runtime console using F2.		
55. Open the 2D Value tab in the Valuators interface. Open the Counter module interface.		
56. Press 'Play' in the Stopwatch.		
57. After START! has appeared, use the Valuator to move the cube towards the blue object.	Notice that every time the yellow cube collides with the blue object the value of the Counter increases by 1.	
Now we want the score to appear on screen so you can see if you hit the object without looking at the Counter. This is easily done by connecting a 3D Text module to the Counter module.		
58. Add a 3D Text module from the module section into the Data Flow Editor.	This creates a 3D Text module displaying the number of collisions that have occurred.	
59. Make a connection between the Counter and the 3D Text module.		
60. Open the 3D Text module's user interface and type: '<v1>' in the text field.		
61. Place the 3D Text in the top left corner of the screen by changing its position (explained in Tutorial #1).		
62. Move the yellow cube and hit the blue object.	Notice the value of the 3D text module increases every time a collision occurs.	
63. Now save the application. 'File | Save Application.	Now your file is saved and can be used in the future.	

Assignment: Try to show fireworks at every collision. Fireworks can be found in the Particle module. Hint: Use the same event as for the Counter module.

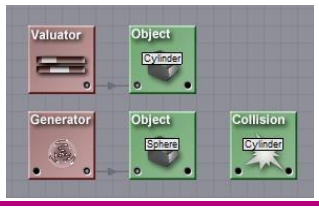
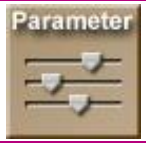
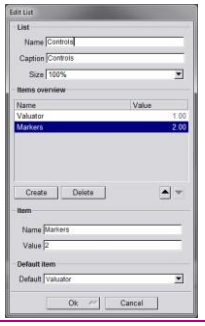
4.4 D-Flow Tutorial #4 Switching sources

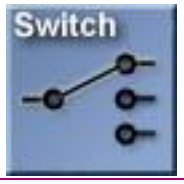
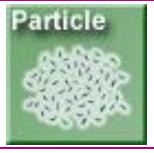
D-Flow Tutorial #4 Switching sources		
Time estimation: 30-40 minutes		
Goal	The goal of this tutorial is to learn how to use events and how to use the Parameter, Switch, Particle and MoCap modules . This is done by creating a small application.	
Topics covered in this tutorial:	• Triggering events based on a specified condition • Triggering events based on collisions between objects • Switching between different input parameters • Creating parameters for the Runtime console	
Modules covered in this tutorial:		**'Parameter'** This module create parameters for in the Runtime console. These parameters could be sliders, check boxes, lists or buttons.
		**'Switch'** This module makes it possible to switch between, for example, different input parameters.
		**'Particle'** This module creates different particles when an event occurs. An example of a particle is the CAREN Fireworks particle.
		**'MoCap'** This module receives all data coming from the motion capture system and the force plates.
References:	D-Flow 3 Help, Module References: Event, Stopwatch, Collision, Valuator and Counter module.	

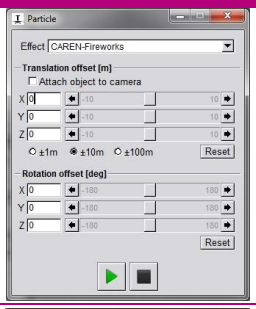
Introduction
As you already have seen it is quite easy to create simple environments with objects that move and collide. This way objects can be used in games as targets, distracters or to create depth in the scene. It is also possible to provide positive feedback when a target is hit. This tutorial extends on the topics of the previous tutorials. An application will be built which can be controlled by different hardware components (markers, valuator or generator). In the Runtime Console the different components can be selected.

Preview of the result after completing this tutorial	
	
Data Flow Editor	**DRS Window**

Procedure	Explanation	Illustration
We will start by creating a scene for this application.		
1. Startup D-Flow and open a new application.		
2. Create two different objects (one red Sphere and one blue Cylinder) and drag the objects to the Data Flow Editor.		
3. Set the transformation of the Sphere to x=0, y=2 and z=-3. Make sure that the Cylinder is positioned in the same plane as the Sphere (so transformation in z=-3).		
4. Create a Collision module and set the collision between the two objects (set the Cylinder as primary object).		
Add controls to the application		
5. Create a Generator module and connect it to the Sphere.		

Procedure	Explanation	Illustration
6. Connect channel 1 to Object.PosX and channel 2 to Object.PosY.		
7. Connect a Valuator module to the Cylinder and set the Valuator settings of parameter 1 and 2 to Min:-10 and Max: 10.		
8. Run the application. Note that when the cursor moves up in the Valuator module, the Cylinder will move down in the DRS window.		
9. Correct the Valuator by inverting the signal from parameter 2 to the Object.PosY with a mathbox.		
10. Run the application again and see that the upwards direction of the cursor is the same as the Cylinder.		
Creating a list in the parameter module		
11. Create a Parameter module and double click on the module to open the User Interface (GUI).		
12. Press Create to create a new List.		
13. Change the Name and Caption to 'Controls'. Create 2 items with the Create button. Change the name of Item 1 to Valuator and Item 2 to Markers. Set the Default item to Valuator and press 'Ok'.		
14. Check the results of the previous step in the Runtime console. A list is created with two options: Valuator and Markers.		

Procedure	Explanation	Illustration
15. Create a Switch module.	A Switch module can be used to switch between multiple input parameters.	
16. Open the user interface and set the mode to 'Many to one' and the Number of sets to 6 and Channels per set also to 6.		
17. Connect the Parameter module to the Switch module and drag a line from Controls to ActiveSet.Index	The number of the list of the controls indicate which channel should be active.	
18. Delete the connection between the Valuator and the Cylinder and make a connection between the Valuator and the Switch.		
19. Drag a line between Parameter 1 to Set1.Channel1 and Parameter 2 to Set1.Channel2. Invert channel 2.		
20. Do not forget to connect the output of the Switch module to the input of the Cylinder.		
21. Drag a MoCap module into the editor window . Open the Mocap user interface and open the Markers tab. Set the number of unlabeled markers to 1.		
22. Create a connection from the MoCap to the Switch module (Marker1.PosX to Set2.Channel1 and Marker1.PosY to Set2.Channel2). In order to do this, you first need to select the channels to output from the Out tab of the MoCap module.		
Now it is time to create the fireworks		
23. Create a global event called 'Collision'.		
24. Open the Collision module's user interface and change the collision events behind True from 'NONE' to 'Collision'.		
25. Create a Particle module and open the user interface.		

Procedure	Explanation	Illustration
26. Set the Effect to CAREN-Fireworks and close the interface.		
27. Open the Event mapping of the Particle module (right mouse click on the module to see the option).	Clicking with the right mouse button reveals the option Edit Event mapping. It is also possible to edit events in the Module Properties window (at the right side of the screen; visible with one click on the specific module).	
28. Set the global event 'Play' to 'NONE' and the Collision from 'NONE' to 'Play'.	CAREN Fireworks will appear when the event Collision occurs.	
29. Test the application. Set the Controls to Valuator on the Default tab and press Play in the Runtime Console. Open the 2D Value tab of the Valuator and move the cursor.	When there is a collision between the Sphere and the Cylinder, the fireworks will be visible.	
Add an extra control for running		
30. Open the Parameter GUI and edit the created list. Add an extra list item and name it Generator.		
31. Drag a (second) generator module in the editor and set channel 1 to: COS(TIME) and channel 2 to: SIN(TIME).		
32. Connect the second generator to the Switch (Set 3.channel1 and 2)		
33. Test the application again, change the Controls in the Runtime Console.	When the Generator is selected in the Runtime Console, you cannot control the application with the Valuator.	
34. Save the application.		

5 Appendix I – Feedback loop components