**Midcities Programming Group**

**Google Dart**

**http://www.meetup.com/Midcities-Programming-Meetup/**

**Contact Jimmy or Caleb from meetup.com**

[**JimmyRuska@gmail.com**](mailto:JimmyRuska@gmail.com) **(jimmyr.com, trainear.com, youtube.jimmyr.com)**

**Past meetups: Android, Haskell, Python, XAMPP**

**More Programming Meetups**

**http://www.tarrantmakers.org/community/**

[**http://www.meetup.com/DFW-Erlang-User-Group/**](http://www.meetup.com/DFW-Erlang-User-Group/events/165679632/)

[**http://www.meetup.com/Startup-Grind-Dallas/**](http://www.meetup.com/Startup-Grind-Dallas/)

**http://www.meetup.com/HTML5-User-Group/**

[**http://www.meetup.com/Tarrant-Makers/**](http://www.meetup.com/Tarrant-Makers/)

[**http://www.meetup.com/DallasFortWorthPUG/**](http://www.meetup.com/DallasFortWorthPUG/)

**http://www.meetup.com/DallasJS/**

[**http://www.meetup.com/dallasmakerspace/**](http://www.meetup.com/dallasmakerspace/)

[**http://www.meetup.com/OpenHack-Denton/**](http://www.meetup.com/OpenHack-Denton/)

**http://www.meetup.com/AngularJS-DFW-Area/**

**http://www.meetup.com/Dallas-R-Users-Group/**

[**http://www.meetup.com/Dallas-Android-Development-Group/**](http://www.meetup.com/Dallas-Android-Development-Group/)

[**http://www.meetup.com/DFW-Nonprofit-Geek-Meetup/**](http://www.meetup.com/DFW-Nonprofit-Geek-Meetup/)

[**http://www.meetup.com/Big-Data-in-the-Big-D/**](http://www.meetup.com/Big-Data-in-the-Big-D/)

[**http://www.meetup.com/dfwpython/**](http://www.meetup.com/dfwpython/)

[**http://www.meetup.com/dallasphp/**](http://www.meetup.com/dallasphp/)

[**http://www.meetup.com/laravel-dallas-fort-worth/**](http://www.meetup.com/laravel-dallas-fort-worth/)

**http://www.meetup.com/Geeknight-Dallas/**

**http://www.meetup.com/Meteor-Dallas/**

**http://www.meetup.com/Fort-Worth-WordPress-Group/**

[**http://www.meetup.com/Fire-Your-Webmaster/**](http://www.meetup.com/Fire-Your-Webmaster/)

[**http://www.meetup.com/DallasNode/**](http://www.meetup.com/DallasNode/)

[**http://www.meetup.com/A-Bunch-of-Short-Guys/**](http://www.meetup.com/A-Bunch-of-Short-Guys/)

[**http://www.meetup.com/WordPress-Midcities-User-Group/**](http://www.meetup.com/WordPress-Midcities-User-Group/)

[**http://www.meetup.com/DFW-WordPress-Developers/**](http://www.meetup.com/DFW-WordPress-Developers/)

[**http://www.meetup.com/dfwwordpress/**](http://www.meetup.com/dfwwordpress/)

[**http://www.meetup.com/joomladallas/**](http://www.meetup.com/joomladallas/)

[**http://www.meetup.com/dallasdrupal/**](http://www.meetup.com/dallasdrupal/)

[**http://www.meetup.com/Dallas-Ruby-on-Rails/**](http://www.meetup.com/Dallas-Ruby-on-Rails/)

**http://www.meetup.com/Dallas-Startup-Happy-Hour/**

[**http://www.dprg.org/**](http://www.dprg.org/)

**http://www.meetup.com/CanJS-Dallas/**

**Get Fast Help:** Get an IRC client and join Freenode, get a stackoverflow and superuser account, search for google groups and google+ programming communities, use our meetup mailing list.

**Google Search:** “Free programming books md”

**Professional Tutorials:** lynda.com, pluralsight, trainsignal, vtc, totaltraining, Tuts+, trainsignal, livelessons, CBT nuggets, Infinite Skills, teach12, educator.com, udemy, skillfeed. Always search youtube too, eg googledevelopers channel

**Google Dart:** <https://www.dartlang.org/tools/download.html>

**Learn Dart:** dartlang.org, try.dartlang.org, “dartlang google+”, “Dartlang Tips”, “What is dart? O’reilly”, “Dart by example”

**What Sucks About JavaScript**

* Subtle differences between different browsers. Your code will break. Libraries add level of abstraction.
* No modules / namespace. You generally load libraries into one big object using immediate functions.
* Dynamic typing makes people obsess over unit tests. The compiler can’t check if you’re correct.
* Many books are written exclusively on javascript quirks. Search “JavaScript Garden” and “JavaScript Patterns”

**Google Dart Solves These Problems**

Dart solves these problems. Dart appeared in 2011 as a language that compiles to javascript or runs through its own virtualmachine. Javascript is a very quirky language that appeared in 1995 for things like simple form validation but has exploded beyond the browser. Dart also adds class based inheritance that OOP programmers are used to, things like optional function arguments, php-style string evaluation, object to array conversion when possible (making it often faster than the same code written in javascript), interfaces, optional typing with typedefs and generics, modules with public and private functions. Dart is also moving towards competing with node.js with the dartvm, which is faster and safer. Angularjs and polymer were ported to dart, allowing a way for people to make single page web applications that do most the rendering on the client side. If you use many libraries, dart uses something called “tree shaking” to only import the code you use.

**Other Languages That Compile to JavaScript:** TypeScript (by Microsoft), livescript, haxe (compiles to many other languages), GWT(Google) and Ceylon (Java), nimrod, CoffeeScript, Elm and fay (haskell-like), clojureScript, js\_of\_ocaml, asm.js and many many more. Google “languages that compile to JS”. Everyone is trying to fix javascript. There is also efforts like the closure compiler which compiles javascript to better javascript. Reinvisioning what javascript should have been is very appealing to most web developers. We don’t want to wait for new versions of ECMAScript.

**Things to Know:**

* Dart can interoperate with libraries written in javascript: <https://www.dartlang.org/articles/js-dart-interop/>
* A “hello world” produces 1000 bytes of javascript. That overhead is relatively static so it becomes less of a problem the bigger your program gets. Very small projects in dart may not be worth the overhead. Typescript does not do this.
* The dartvm isn’t going to be released in all browsers anytime soon. Generally you’re going to be compiling dart into javascript. You can still use dart as a general purpose language. It can run as a server or interact with the filesystem if you use it from the dartvm.
* Dart Editor is a repurposed version of eclipse that comes with dart. The command line dart tools are in ./dart-sdk/bin/

**Compile From the Command Line:**

|  |  |
| --- | --- |
| // foo.dart, hello world example, this is a comment  main() { print('Hello, Dart! This will print in console.'); } | .\dart\dart-sdk\bin\dart2js --minify -o hello.js foo.dart  Makes a file called hello.js. Embedding it prints Hello Dart |

**Comparison:**

|  |  |
| --- | --- |
| **JavaScript** | **Dart** |
| Math.floor(5/2); | 5 ~/ 2 |
| function takeHalf(x){ return x/2; } | takeHalf (x) => x/2 |
| var someString=”foo”; | var someString =”foo”;  String someString = “foo”;  final String someString = “foo”; (final = immutable) |
| var a=1; console.log("I have "+a.toString()+" apples");  var a=Array(1,2,3); console.log(a[0]+‘ apples’); | var a=1; print(‘I have $a apples’);  List a=[1,2,3]; print(‘${a[0]} apples’); |
| Math.floor(3.2); | 3.2.floor(); |
| ‘Hello’ + ‘World’ = “HelloWorld” | “Hello” “World” = “HelloWorld” (Don’t need the +) |
| var x; // x is undefined… and undefined is a global var | var x; // x is null, void functions also return null |
| if (x) …x will be treated as false if the value is false, NaN, null, undefined, “”, 0, otherwise, it is true | if (x) .. checks if x is the bool value true |
| function Person(first, last){ this.first; this.last; }  var dude = new Person; | class Person { String first; String last;  Person(this.first, this.last); }  Person dude = new Person(); |

**Dart Tips**

|  |  |
| --- | --- |
| Declare a library | library myLibrary; |
| Import a library | import “dart:html”; |
| Strings that allow linebreaks | “””Use triple quotes to have a  string that contains linebreaks“”” |
| Raw Strings | r“Use raw strings and you will be able to see this \n” |
| Dart Types | **Numbers**: num, int, double, **Strings**: String, StringBuffer, RegExp, **Booleans**: bool, true, false, **Dates & Times**: DateTime, Duration, Stopwatch, **Collections**: List<E>, Set<E>, Map<K, V>, Queue |
| Query the DOM with css selectors | querySelector(Selector), querySelectorAll(Selector)  (query() and queryAll() got deprecated) |
| Optional Arguments: positional and named  You can pass optional args which will default to a value if none is set. You can also pass properties to the function by name as if they were properties of an object. | sayName(first, [last=”doe”], {middle: ‘’}){  print(‘Hi $first $middle $last!’);  }  sayName(‘jimmy’); // will print “Hi Jimmy Doe”  sayName(‘Jimmy’, ‘Ruska’); will print “Hi Jimmy Ruska”  sayname(‘Jimmy’, ‘Ruska’, middle: ‘Chris’); // Hi Jimmy C. R. |
| Setting multiple properties of the same object  -- querySelector(x) returns a html element object, then we use “..” to access its properties “text” and “onClick” | querySelector("#sample\_text\_id")  ..text = "Click me!"  ..onClick.listen(reverseText); |

**Tutorial -- Meetup Tutorial Starts Here**

Click File -> New Application -> name it “meetup” and click finish. Click ctrl+r. Dartium should have loaded up. Dartium is chromium with the dartvm included. If you click “click me” it will reverse the text.

|  |  |
| --- | --- |
| **Delete everything in meetup.dart except**  import 'dart:html';  void main() {    } | **Delete this from meetup.html**  <div id="sample\_container\_id"> <p id="sample\_text\_id"> Click me!</p></div>  **Replace it with:**  Double: <div id=”my\_container”></div>  <input type=”text” id=”my\_input” value=”5”/>  <input type=”button” id=”my\_button” value=”Double!”/> |

Click ctrl+r. We can see code we added creates an input box and a button. Next put this between main’s curly braces

*print(querySelector('#my\_input').value);*

Then click ctrl+r. Close dartium and look at the console under the source code pane. It should say 5. This is because in our input box we put value=”5”.

Lets add a print statement for when you click on things. Put this inside main.

querySelector('#my\_button').onClick.listen((event) => print(event) );

This adds adds a mouseclick listener to the element whose id=”my\_button”. Every time you click it, it will pass an event object into our custom function, which just prints it. Click ctrl+r, click the “Double!” button a few times, close dartium, and look at the console. It should show many mouse events. Lets change it to this:

querySelector('#my\_button').onClick.listen((event) => window.console.debug(event) );

Run the code. When dartium opens, click ctrl+shift+j, then select the console tab if it’s not already selected. Click the “double!” button a few times. You can click to expand the events that appear in the console and see what’s inside the event object we had printed before. window.console.debug can be much more useful than print at times. Delete what you have inside “main” and lets replace it with this.

querySelector('#my\_button').onClick.listen((event){

String input=querySelector("#my\_input").value;

window.alert(input);

});

This will short an alert box with the value of the input box with id=“my\_input” every time the “double!” button is clicked. Notice I changed the anonymous function to (event){...} instead of (event) => …. The brackets are for when we want to span multiple lines. The “=> ...” is pretty much the same thing as “{return ..}” which shortens a lot of common expressions. Remove window.alert(input); and replace it with

querySelector("#my\_container").text=(int.parse(input)\*2).toString();

Run the code and click the button. This parses input into an int, then multiplies it by 2, then converts it to a string. The resulting value is put inside the div element with id=”my\_container”. Our double function works! Try changing the “5” to “51” and click it again.

What does the “#” in “#my\_container” mean? These are css selectors you can use to query the dom. jQuery’s website has several interactive examples search “jquery selectors”. Search “css selector cheat sheet”. Here are a few translations:

|  |  |
| --- | --- |
| div#foobar.moo | select all div elements, then of those elements, select the ones with the ID “foobar”, then of those elements, select the ones with the class “moo” |
| div#foobar.moo > p span | Grab the elements from the above, then select the ones that have a paragraph tag as first children, then recursively search the paragraph tag for any span tags |

If we have any more time let’s try the tutorials on the dart site: https://www.dartlang.org/codelabs/darrrt/