###$$$$##$####$$$$$##########

=========================JQUERY=========================================

##### Q1. What is jQuery?

Ans: jQuery is fast, lightweight and feature-rich client side JavaScript Library/Framework which helps in to traverse HTML DOM, make animations, add Ajax interaction, manipulate the page content, change the style and provide cool UI effect. It is one of the most popular client side library and as per a survey it runs on every second website.

Q2. Why do we use jQuery?

Ans: Due to following advantages.

Easy to use and learn.

Easily expandable.

Cross-browser support (IE 6.0+, FF 1.5+, Safari 2.0+, Opera 9.0+)

Easy to use for DOM manipulation and traversal.

Large pool of built in methods.

AJAX Capabilities.

Methods for changing or applying CSS, creating animations.

Event detection and handling.

Tons of plug-ins for all kind of needs.

Q3. How JavaScript and jQuery are different?

Ans: JavaScript is a language While jQuery is a library built in the JavaScript language that helps to use the JavaScript language.

Q4. Is jQuery replacement of Java Script?

Ans: No. jQuery is not a replacement of JavaScript. jQuery is a different library which is written on top of JavaScript. jQuery is a lightweight JavaScript library that emphasizes interaction between JavaScript and HTML.

Q5. Is jQuery a library for client scripting or server scripting?

Ans. Client side scripting.

Q6. Is jQuery a W3C standard?

Ans: No. jQuery is not a W3C standard.

Q7. What is the basic need to start with jQuery?

Ans: To start with jQuery, one need to make reference of it's library. The latest version of jQuery can be downloaded from jQuery.com.

Q8. Which is the starting point of code execution in jQuery?

Ans: The starting point of jQuery code execution is $(document).ready() function which is executed when DOM is loaded.

Q9. What does dollar sign ($) means in jQuery?

Ans: Dollar Sign is nothing but it's an alias for JQuery. Take a look at below jQuery code.

Collapse | Copy Code

$(document).ready(function(){

});

Over here $ sign can be replaced with "jQuery" keyword.

Collapse | Copy Code

jQuery(document).ready(function(){

});

Q10. Can we have multiple document.ready() function on the same page?

Ans: YES. We can have any number of document.ready() function on the same page.

Q11. Can we use our own specific character in the place of $ sign in jQuery?

Ans: Yes. It is possible using jQuery.noConflict().

Q12. Is it possible to use other client side libraries like MooTools, Prototype along with jQuery?

Ans: Yes.

Q13. What is jQuery.noConflict?

Ans: As other client side libraries like MooTools, Prototype can be used with jQuery and they also use $() as their global function and to define variables. This situation creates conflict as $() is used by jQuery and other library as their global function. To overcome from such situations, jQuery has introduced jQuery.noConflict().

Collapse | Copy Code

jQuery.noConflict();

// Use jQuery via jQuery(...)

jQuery(document).ready(function(){

jQuery("div").hide();

});

You can also use your own specific character in the place of $ sign in jQuery.

Collapse | Copy Code

var $j = jQuery.noConflict();

// Use jQuery via jQuery(...)

$j(document).ready(function(){

$j("div").hide();

});

Q14. Is there any difference between body onload() and document.ready() function?

Ans: document.ready() function is different from body onload() function for 2 reasons.

We can have more than one document.ready() function in a page where we can have only one body onload function.

document.ready() function is called as soon as DOM is loaded where body.onload() function is called when everything gets loaded on the page that includes DOM, images and all associated resources of the page.

Q15. What is the difference between .js and .min.js?

Ans: jQuery library comes in 2 different versions Production and Deployment. The deployment version is also known as minified version. So .min.js is basically the minified version of jQuery library file. Both the files are same as far as functionality is concerned. but .min.js is quite small in size so it loads quickly and saves bandwidth.

Q16. Why there are two different version of jQuery library?

Ans: jQuery library comes in 2 different versions.

Production

Deployment

The production version is quite useful at development time as jQuery is open source and if you want to change something then you can make those changes in production version. But the deployment version is minified version or compressed version so it is impossible to make changes in it. Because it is compressed, so its size is very less than the production version which affects the page load time.

Q17. What is a CDN?

Ans: A content delivery network or content distribution network (CDN) is a large distributed system of servers deployed in multiple data centers across the Internet. The goal of a CDN is to serve content to end-users with high availability and high performance.

Q18. Which are the popular jQuery CDN? and what is the advantage of using CDN?

Ans: There are 3 popular jQuery CDNs.

1. Google.

2. Microsoft

3. jQuery.

Advantage of using CDN.

It reduces the load from your server.

It saves bandwidth. jQuery framework will load faster from these CDN.

The most important benefit is it will be cached, if the user has visited any site which is using jQuery framework from any of these CDN

Q19. How to load jQuery from CDN?

Ans: Below is the code to load jQuery from all 3 CDNs.  
Code to load jQuery Framework from Google CDN

Collapse | Copy Code

<script type="text/javascript"

src="http://ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js">

</script>

Code to load jQuery Framework from Microsoft CDN

Collapse | Copy Code

<script type="text/javascript"

src="http://ajax.microsoft.com/ajax/jquery/jquery-1.9.1.min.js">

</script>

Code to load jQuery Framework from jQuery Site(EdgeCast CDN)

Collapse | Copy Code

<script type="text/javascript"

src="http://code.jquery.com/jquery-1.9.1.min.js">

</script>

Q20. How to load jQuery locally when CDN fails?

Ans: It is a good approach to always use CDN but sometimes what if the CDN is down (rare possibility though) but you never know in this world as anything can happen.  
  
Below given jQuery code checks whether jQuery is loaded from Google CDN or not, if not then it references the jQuery.js file from your folder.

Collapse | Copy Code

<script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js"></script>

<script type="text/javascript">

if (typeof jQuery == 'undefined')

{

document.write(unescape("%3Cscript src='Scripts/jquery.1.9.1.min.js' type='text/javascript'%3E%3C/script%3E"));

}

</script>

It first loads the jQuery from Google CDN and then check the jQuery object. If jQuery is not loaded successfully then it will references the jQuery.js file from hard drive location. In this example, the jQuery.js is loaded from Scripts folder.

Q21. What are selectors in jQuery and how many types of selectors are there?

Ans: To work with an element on the web page, first we need to find them. To find the html element in jQuery we use selectors. There are many types of selectors but basic selectors are:

Name: Selects all elements which match with the given element Name.

#ID: Selects a single element which matches with the given ID

.Class: Selects all elements which match with the given Class.

Universal (\*): Selects all elements available in a DOM.

Multiple Elements E, F, G: Selects the combined results of all the specified selectors E, F or G.

Attribute Selector: Select elements based on its attribute value.

Q22. How do you select element by ID in jQuery?

Ans: To select element use ID selector. We need to prefix the id with "#" (hash symbol). For example, to select element with ID "txtName", then syntax would be,

Collapse | Copy Code

$('#txtName')

Q23. What does $("div") will select?

Ans: This will select all the div elements on page.

Q24. How to select element having a particular class (".selected")?

Ans: $('.selected'). This selector is known as class selector. We need to prefix the class name with "." (dot).

Q25. What does $("div.parent") will select?

Ans: All the div element with parent class.

Q26. What are the fastest selectors in jQuery?

Ans: ID and element selectors are the fastest selectors in jQuery.

Q27. What are the slow selectors in jQuery?

Ans: class selectors are the slow compare to ID and element.

Q28. How jQuery selectors are executed?

Ans: Your last selectors is always executed first. For example, in below jQuery code, jQuery will first find all the elements with class ".myCssClass" and after that it will reject all the other elements which are not in "p#elmID".

Collapse | Copy Code

$("p#elmID .myCssClass");

Q29. Which is fast document.getElementByID('txtName') or $('#txtName').?

Ans: Native JavaScipt is always fast. jQuery method to select txtName "$('#txtName')" will internally makes a call to document.getElementByID('txtName'). As jQuery is written on top of JavaScript and it internally uses JavaScript only So JavaScript is always fast.

Q30. Difference between $(this) and 'this' in jQuery?

Ans: this and $(this) refers to the same element. The only difference is the way they are used. 'this' is used in traditional sense, when 'this' is wrapped in $() then it becomes a jQuery object and you are able to use the power of jQuery.

Collapse | Copy Code

$(document).ready(function(){

$('#spnValue').mouseover(function(){

alert($(this).text());

});

});

In below example, this is an object but since it is not wrapped in $(), we can't use jQuery method and use the native JavaScript to get the value of span element.

Collapse | Copy Code

$(document).ready(function(){

$('#spnValue').mouseover(function(){

alert(this.innerText);

});

});

Q31. How do you check if an element is empty?

Ans: There are 2 ways to check if element is empty or not. We can check using ":empty" selector.

Collapse | Copy Code

$(document).ready(function(){

if ($('#element').is(':empty')){

//Element is empty

}

});

And the second way is using the "$.trim()" method.

Collapse | Copy Code

$(document).ready(function(){

if($.trim($('#element').html())=='') {

//Element is empty

}

});

Q32. How do you check if an element exists or not in jQuery?

Ans: Using jQuery length property, we can ensure whether element exists or not.

Collapse | Copy Code

$(document).ready(function(){

if ($('#element').length > 0){

//Element exists

});

});

Q33. What is the use of jquery .each() function?

Ans: The $.each() function is used to iterate over a jQuery object. The $.each() function can be used to iterate over any collection, whether it is an object or an array.

Q34. What is the difference between jquery.size() and jquery.length?

Ans: jQuery .size() method returns number of element in the object. But it is not preferred to use the size() method as jQuery provide .length property and which does the same thing. But the .length property is preferred because it does not have the overhead of a function call.

Q35. What is the difference between $('div') and $('<div/>') in jQuery?

Ans: $('<div/>') : This creates a new div element. However this is not added to DOM tree unless you don't append it to any DOM element.  
  
$('div') : This selects all the div element present on the page.

Q36. What is the difference between parent() and parents() methods in jQuery?

Ans: The basic difference is the parent() function travels only one level in the DOM tree, where parents() function search through the whole DOM tree.

Q37. What is the difference between eq() and get() methods in jQuery?

Ans: eq() returns the element as a jQuery object. This method constructs a new jQuery object from one element within that set and returns it. That means that you can use jQuery functions on it.  
  
get() return a DOM element. The method retrieve the DOM elements matched by the jQuery object. But as it is a DOM element and it is not a jQuery-wrapped object. So jQuery functions can't be used. Find out more here.

Q38. How do you implement animation functionality?

Ans: The .animate() method allows us to create animation effects on any numeric CSS property. This method changes an element from one state to another with CSS styles. The CSS property value is changed gradually, to create an animated effect.  
  
Syntax is:

Collapse | Copy Code

(selector).animate({styles},speed,easing,callback)

styles: Specifies one or more CSS properties/values to animate.

duration: Optional. Specifies the speed of the animation.

easing: Optional. Specifies the speed of the element in different points of the animation. Default value is "swing".

callback: Optional. A function to be executed after the animation completes.

Simple use of animate function is,

Collapse | Copy Code

$("btnClick").click(function(){

$("#dvBox").animate({height:"100px"});

});

Q39. How to disable jQuery animation?

Ans: Using jQuery property "jQuery.fx.off", which when set to true, disables all the jQuery animation. When this is done, all animation methods will immediately set elements to their final state when called, rather than displaying an effect.

Q40. How do you stop the currently-running animation?

Ans: Using jQuery ".stop()" method.

Q41. What is the difference between .empty(), .remove() and .detach() methods in jQuery?

Ans: All these methods .empty(), .remove() and .detach() are used for removing elements from DOM but they all are different.  
  
.empty(): This method removes all the child element of the matched element where remove() method removes set of matched elements from DOM.  
  
.remove(): Use .remove() when you want to remove the element itself, as well as everything inside it. In addition to the elements themselves, all bound events and jQuery data associated with the elements are removed.   
  
.detach(): This method is the same as .remove(), except that .detach() keeps all jQuery data associated with the removed elements. This method is useful when removed elements are to be reinserted into the DOM at a later time.   
  
Find out more here

Q42. Explain .bind() vs .live() vs .delegate() vs .on()

Ans: All these 4 jQuery methods are used for attaching events to selectors or elements. But they all are different from each other.  
  
.bind(): This is the easiest and quick method to bind events. But the issue with bind() is that it doesn't work for elements added dynamically that matches the same selector. bind() only attach events to the current elements not future element. Above that it also has performance issues when dealing with a large selection.  
  
.live(): This method overcomes the disadvantage of bind(). It works for dynamically added elements or future elements. Because of its poor performance on large pages, this method is deprecated as of jQuery 1.7 and you should stop using it. Chaining is not properly supported using this method.  
  
.delegate(): The .delegate() method behaves in a similar fashion to the .live() method, but instead of attaching the selector/event information to the document, you can choose where it is anchored and it also supports chaining.  
  
.on(): Since live was deprecated with 1.7, so new method was introduced named ".on()". This method provides all the goodness of previous 3 methods and it brings uniformity for attaching event handlers.  
  
Find out more here

Q43. What is wrong with this code line "$('#myid.3').text('blah blah!!!');"

Ans: The problem with above statement is that the selectors is having meta characters and to use any of the meta-characters ( such as !"#$%&'()\*+,./:;<=>?@[\]^`{|}~ ) as a literal part of a name, it must be escaped with with two backslashes: \\. For example, an element with id="foo.bar", can use the selector $("#foo\\.bar").   
So the correct syntax is,

Collapse | Copy Code

$('#myid\\.3').text('blah blah!!!');

Q44. How to create clone of any object using jQuery?

Ans: jQuery provides clone() method which performs a deep copy of the set of matched elements, meaning that it copies the matched elements as well as all of their descendant elements and text nodes.

Collapse | Copy Code

$(document).ready(function(){

$('#btnClone').click(function(){

$('#dvText').clone().appendTo('body');

return false;

});

});

Q45. Does events are also copied when you clone any element in jQuery?

Ans: As explained in previous question, using clone() method, we can create clone of any element but the default implementation of the clone() method doesn't copy events unless you tell the clone() method to copy the events. The clone() method takes a parameter, if you pass true then it will copy the events as well.

Collapse | Copy Code

$(document).ready(function(){

$("#btnClone").bind('click', function(){

$('#dvClickme').clone(true).appendTo('body');

});

​

Q46. What is difference between prop and attr?

Ans: attr(): Get the value of an attribute for the first element in the set of matched elements. Whereas,.prop(): (Introduced in jQuery 1.6) Get the value of a property for the first element in the set of matched elements.   
  
Attributes carry additional information about an HTML element and come in name="value" pairs. Where Property is a representation of an attribute in the HTML DOM tree. once the browser parse your HTML code ,corresponding DOM node will be created which is an object thus having properties.   
  
attr() gives you the value of element as it was defines in the html on page load. It is always recommended to use prop() to get values of elements which is modified via javascript/jquery , as it gives you the original value of an element's current state. Find out more here.

Q47. What is event.PreventDefault?

Ans: The event.preventDefault() method stops the default action of an element from happening. For example, Prevents a link from following the URL.

Q48. What is the difference between event.PreventDefault and event.stopPropagation?

Ans: event.preventDefault(): Stops the default action of an element from happening.   
event.stopPropagation(): Prevents the event from bubbling up the DOM tree, preventing any parent handlers from being notified of the event. For example, if there is a link with a click method attached inside of a DIV or FORM that also has a click method attached, it will prevent the DIV or FORM click method from firing.

Q49. What is the difference between event.PreventDefault and "return false"?

Ans: e.preventDefault() will prevent the default event from occurring, e.stopPropagation() will prevent the event from bubbling up and return false will do both.

Q50. What is the difference between event.stopPropagation and event.stopImmediatePropagation?

Ans: event.stopPropagation() allows other handlers on the same element to be executed, while event.stopImmediatePropagation() prevents every event from running. For example, see below jQuery code block.

Collapse | Copy Code

$("p").click(function(event){

event.stopImmediatePropagation();

});

$("p").click(function(event){

// This function won't be executed

$(this).css("background-color", "#f00");

});

If event.stopPropagation was used in previous example, then the next click event on p element which changes the css will fire, but in case event.stopImmediatePropagation(), the next p click event will not fire.

Q51. How to check if number is numeric while using jQuery 1.7+?

Ans: Using "isNumeric()" function which was introduced with jQuery 1.7.

Q52. How to check data type of any variable in jQuery?

Ans: Using $.type(Object) which returns the built-in JavaScript type for the object.

Q53. How do you attach a event to element which should be executed only once?

Ans: Using jQuery one() method. This attaches a handler to an event for the element. The handler is executed at most once per element. In simple terms, the attached function will be called only once.

Collapse | Copy Code

$(document).ready(function() {

$("#btnDummy").one("click", function() {

alert("This will be displayed only once.");

});

});​

Q54. Can you include multiple version of jQuery? If yes, then how they are executed?

Ans: Yes. Multiple versions of jQuery can be included in same page.

Q55. In what situation you would use multiple version of jQuery and how would you include them?

Ans: Well, it is quite possible that the jQuery plugins which are used are dependent on older version but for your own jQuery code, you would like to use newer version. So because of this dependency, multiple version of jQuery may required sometimes on single page.  
  
Below code shows how to include multiple version of jQuery.

Collapse | Copy Code

<script type='text/javascript' src='js/jquery\_1.9.1.min.js'></script>

<script type='text/javascript'>

var $jq = jQuery.noConflict();

</script>

<script type='text/javascript' src='js/jquery\_1.7.2.min.js'></script>

By this way, for your own jQuery code use "$jq", instead of "$" as "$jq" refers to jQuery 1.9.1, where "$" refers to 1.7.2.

Q56. Is it possible to hold or delay document.ready execution for sometime?

Ans: Yes, its possible. With Release of jQuery 1.6, a new method "jQuery.holdReady(hold)" was introduced. This method allows to delay the execution of document.ready() event. document.ready() event is called as soon as your DOM is ready but sometimes there is a situation when you want to load additional JavaScript or some plugins which you have referenced.

Collapse | Copy Code

​

$.holdReady(true);

$.getScript("myplugin.js", function() {

$.holdReady(false);

});

Q57. What is chaining in jQuery?

Ans: Chaining is one of the most powerful feature of jQuery. In jQuery, Chaining means to connect multiple functions, events on selectors. It makes your code short and easy to manage and it gives better performance. The chain starts from left to right. So left most will be called first and so on.

Collapse | Copy Code

​$(document).ready(function(){

$('#dvContent').addClass('dummy');

$('#dvContent').css('color', 'red');

$('#dvContent').fadeIn('slow');

});​

The above jQuery code sample can be re-written using chaining. See below.

Collapse | Copy Code

​$(document).ready(function(){

$('#dvContent').addClass('dummy')

.css('color', 'red')

.fadeIn('slow');

});​

Not only functions or methods, chaining also works with events in jQuery. Find out more here.

Q58. How does caching helps and how to use caching in jQuery?

Ans: Caching is an area which can give you awesome performance, if used properly and at the right place. While using jQuery, you should also think about caching. For example, if you are using any element in jQuery more than one time, then you must cache it. See below code.

Collapse | Copy Code

$("#myID").css("color", "red");

//Doing some other stuff......

$("#myID").text("Error occurred!");

​

Now in above jQuery code, the element with #myID is used twice but without caching. So both the times jQuery had to traverse through DOM and get the element. But if you have saved this in a variable then you just need to reference the variable. So the better way would be,

Collapse | Copy Code

var $myElement = $("#myID").css("color", "red");

//Doing some other stuff......

$myElement.text("Error occurred!");

​

So now in this case, jQuery won't need to traverse through the whole DOM tree when it is used second time. So in jQuery, Caching is like saving the jQuery selector in a variable. And using the variable reference when required instead of searching through DOM again.

Q59. You get "jquery is not defined" or "$ is not defined" error. What could be the reason?

Ans: There could be many reasons for this.

You have forgot to include the reference of jQuery library and trying to access jQuery.

You have include the reference of the jQuery file, but it is after your jQuery code.

The order of the scripts is not correct. For example, if you are using any jQuery plugin and you have placed the reference of the plugin js before the jQuery library then you will face this error.

Find out more here.

Q60. How to write browser specific code using jQuery?

Ans: Using jQuery.browser property, we can write browser specific code. This property contains flags for the useragent, read from navigator.userAgent. This property was removed in jQuery 1.9.

Q61. Can we use jQuery to make ajax request?

Ans: Yes. jQuery can be used for making ajax request.

Q62. What are various methods to make ajax request in jQuery?

Ans: Using below jQuery methods, you can make ajax calls.

load() : Load a piece of html into a container DOM

$.getJSON(): Load JSON with GET method.

$.getScript(): Load a JavaScript file.

$.get(): Use to make a GET call and play extensively with the response.

$.post(): Use to make a POST call and don't want to load the response to some container DOM.

$.ajax(): Use this to do something on XHR failures, or to specify ajax options (e.g. cache: true) on the fly.

Find out more here.

Q63. Is there any advantage of using $.ajax() for ajax call against $.get() or $.post()?

Ans: By using jQuery post()/ jQuery get(), you always trust the response from the server and you believe it is going to be successful all the time. Well, it is certainly not a good idea to trust the response. As there can be n number of reason which may lead to failure of response.   
  
Where jQuery.ajax() is jQuery's low-level AJAX implementation. $.get and $.post are higher-level abstractions that are often easier to understand and use, but don't offer as much functionality (such as error callbacks). Find out more here.

Q64. What are deferred and promise object in jQuery?

Ans: Deferred and promise are part of jQuery since version 1.5 and they help in handling asynchronous functions like Ajax. Find out more here.

Q65. Can we execute/run multiple Ajax request simultaneously in jQuery? If yes, then how?

Ans: Yes, it is possible to execute multiple Ajax request simultaneously or in parallel. Instead of waiting for first ajax request to complete and then issue the second request is time consuming. The better approach to speed up things would be to execute multiple ajax request simultaneously.  
  
Using jQuery .when() method which provides a way to execute callback functions based on one or more objects, usually Deferred objects that represent asynchronous events. Find out more here.

Q66. Can you call C# code-behind method using jQuery? If yes,then how?

Ans: Yes. We can call C# code-behind function via $.ajax. But for do that it is compulsory to mark the method as WebMethod.

Q67. Which is the latest version of jQuery library?

Ans: The latest version (when this post is written) of jQuery is 1.10.2 or 2.0.3. jQuery 2.x has the same API as jQuery 1.x, but does not support Internet Explorer 6, 7, or 8.

Q68. Does jQuery 2.0 supports IE?

Ans: No. jQuery 2.0 has no support for IE 6, IE 7 and IE 8.

Q69. What are source maps in jQuery?

Ans: In case of jQuery, Source Map is nothing but mapping of minified version of jQuery against the un-minified version. Source map allows to debug minified version of jQuery library. Source map feature was release with jQuery 1.9. Find out more here.

Q70. How to use migrate jQuery plugin?

Ans: with release of 1.9 version of jQuery, many deprecated methods were discarded and they are no longer available. But there are many sites in production which are still using these deprecated features and it's not possible to replace them overnight. So jQuery team provided with jQuery Migrate plugin that makes code written prior to 1.9 work with it.  
  
So to use old/deprecated features, all you need to do is to provide reference of jQuery Migrate Plugin. Find out more here.

Q71. Is it possible to get value of multiple CSS properties in single statement?

Ans: Well, before jQuery 1.9 release it was not possible but one of the new feature of jQuery 1.9 was .css() multi-property getter.

Collapse | Copy Code

var propCollection = $("#dvBox").css([ "width", "height", "backgroundColor" ]);

In this case, the propCollection will be an array and it will look something like this.

Collapse | Copy Code

{

width: "100px",

height: "200px",

backgroundColor: "#FF00FF"

}

Q72. How do you stop the currently-running animation, remove all queued animations, and complete all animations for the matched elements?

Ans: It can be done via calling .stop([clearQueue ] [, jumpToEnd ]) method and by passing both the parameters as true.

Q73. What is finish method in jQuery?

Ans: The .finish() method stops all queued animations and places the element(s) in their final state. This method was introduced in jQuery 1.9.

Q74. What is the difference between calling stop(true,true) and finish method?

Ans: The .finish() method is similar to .stop(true, true) in that it clears the queue and the current animation jumps to its end value. It differs, however, in that .finish() also causes the CSS property of all queued animations to jump to their end values, as well.

Q75. Consider a scenario where things can be done easily with javascript, would you still prefer jQuery?

Ans: No. If things can be done easily via CSS or JavaScript then You should not think about jQuery. Remember, jQuery library always comes with xx kilobyte size and there is no point of wasting bandwidth.

Q76. Can we use protocol less URL while referencing jQuery from CDNs?

Ans: Yes. Below code is completely valid.

Collapse | Copy Code

<script type="text/javascript" src="//ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js"></script>

Q77. What is the advantage of using protocol less URL while referencing jQuery from CDNs?

Ans: It is quite useful when you are moving from HTTP to HTTPS url. You need to make sure that correct protocol is used for referencing jQuery library as pages served via SSL should contain no references to content served through unencrypted connections.  
  
"protocol-less" URL is the best way to reference third party content that’s available via both HTTP and HTTPS. When a URL’s protocol is omitted, the browser uses the underlying document’s protocol instead. Find out more here.

Q78. What is jQuery plugin and what is the advantage of using plugin?

Ans: A plug-in is piece of code written in a standard JavaScript file. These files provide useful jQuery methods which can be used along with jQuery library methods. jQuery plugins are quite useful as its piece of code which is already written by someone and re-usable, which saves your development time.

Q79. What is jQuery UI?

Ans: jQuery UI is a curated set of user interface interactions, effects, widgets, and themes built on top of the jQuery JavaScript Library that can be used to build interactive web applications.

Q80. What is the difference between jQuery and jQuery UI?

Ans: jQuery is the core library. jQueryUI is built on top of it. If you use jQueryUI, you must also include jQuery.

#################

1.questions on guthub

2.what is single table inheritance and example

3.what is polymorphic association and example

Rails 3 polymorphic associations explained (somewhat)

So, in past few days, I got requests from people to offer a Quiz type question at PollDaal. Being a nice fellow as I am (Modesty at its best!), I listened to them and started looking for ways to do it elegantly in Rails 3. After searching for a bit here and there, I got to know about polymorphic associations could be my saviour. I had learned earlier what they were but that was that. I quickly forgot them and thus they never came in my mind. So, someone said, teaching stuff makes you remember it. Thus, here I present, Polymorphic Associations 101 (kinda).

First things first, what are polymorphic associations?

Well, With polymorphic associations, a model can belong to more than one other model, on a single association. For example, you might have a picture model that belongs to either an employee model or a product model (visualized in the pic above). Don’t worry if you don’t understand the code written there. You will after reading this article (probably).

The most weird part of the above code seems to be the part which says

belongs\_to :imageable, :polymorphic => true

This just means that the model Picture does not belong to any of the Employee or Product model but to something called Imageable.

Uh, oh. I was taught that models belong to models and nothing else. You can check any code which has belongs\_to clause.

Well, that is kinda true, but we form a little twist while using polymorphic associations (if you forgot, they mean that it could be associated with many models). So, imageable here acts as an interface to link to for other models which link the same way they did except now they add a clause of as: :imageable. For example, the Person model now looks like:

class Person < ActiveRecord::Base

has\_many :pictures, :as => :imageable

end

You could have used any word in place of imageable, like picturable, photogenic, photograph, baklind, dhoni anything! They just have to match your column name.

Wait, what? Which column? What column? Which table?

Good question. You have to setup a reference in your Picture model to store the foreign key of the row it ‘belongs\_to’. Now how do we do that? First, let us look at how we did for non-polymorphic associations. If, for example we have a Question model and Choice model, we say choice belongs to question and question has many choices. So, to make that thing work properly, we generally include a column ‘question\_id’ in the ‘choices’ table. That thing generally works out without any problem and we live a happy satisfied life. But, there is no specific model assigned here in the polymorphic associations. What field do we use then? A wise man said, double the amount if you are not sure. Lets follow that wise man. We use two foreign keys! For the Picture case, they foreign keys are ‘imageable\_id’ and ‘imageable\_type’. The migration looks something like this,

add\_column :pictures, :imageable\_id, :integer

add\_column :pictures, :imageable\_type, :string

or you can use a handy trick provided by rails,

t.references :imageable, :polymorphic => true

Just run the migration now, add the snippets to your model, and congratulations. You successfully have configured a polymorphic association!

Yayy!! Well that was simple. Thanks man. Now, how do I actually reference it?

Ahh, the easy part. Let me assume that you know how to get data from model ( person = Person.find(23) ). There are two ways to create a reference, from the parent side or from child side.

person.picture = picture

or

picture.imageable = person

That is all really is there to assigning. Don’t forget to call save method on picture though, otherwise its gone as soon as the object stops persisting.

Seems simple enough, but I have a question. How do you use it to accept data from forms?

This is tricky and so I think I should leave it to experts. Check out this Railscasts. It shows how to do all this for comments. Ideal if you wish to create facebook and would need comments for all your Photo, Status, Video models.

You can either use the method shown by Ryan Bates in Railscasts or do manual assigning the way I showed. I prefer doing the manual assigning though. Easier to setup and less confusing. Plus, you have full control as you know where the code is and what it does.

I have one little trick which I’d like to share. As many people know, adding index to column in tables makes it for faster searching for data through that column, you all will be tempted to add index to both imageable\_type and imageable\_id columns by typing two migrations like this

add\_index :pictures, :imageable\_id

add\_index :pictures, :imageable\_type

And now, you should have faster finds, shouldn’t you? Well no! What we did was create two different indexes in the pictures table but we are always going to search for them together, instead we should do

add\_index :comments, [:commentable\_type, :commentable\_id]

While we are at it, we can also limit the size of commentable\_type column. We are storing it as a string but do you think we would really have a model name of 255 characters? Decreasing the size will also allow for smaller indexing and faster searching. We can limit the size by passing an option

:limit => 15

4.named scope

5.difference between instance\_eval and class\_eval

6.what blocks in rails

7.difference between rails 2 and rails3

8.difference between rails3 and rails4

9.fetchind second highest record from table

10.meta programming techniques in ruby

11.method overloading and method overriding in ruby

12.different ways of calling a method in ruby

13.differece between module and call in ruby

14.mixins concept in ruby

15.helpers

16.observers in rsils

17.caching in rails

18.filters and callbacks in rails

19.testing in rails mock test smoke test

20.how to upload files in ruby

21.updatind multiples models using single form

22.eager loading and lazy loading

23.include and extend difference

24.proc and lamda difference

25. methid arguments & and \*

26.difference between private public and protrcted

27.implementing searching criteria in rails.

28.string and symbols digference

29. multiple methods with same name in rails

30.interpolation in rsils

31.deployment process how to deply a rails application

32.difference between habtm and has\_many through

33.explsin mvc architecture

34.number and cillection difference

35.render and redirect difference

36.delegate method usage

37.how to test a private method in rails

38.types testing in raild

39.how to test a controller sction in rails

40.what is irb

41.difference bet gemfile and gemfile.lock

42.what is bundler

43.load require autoload relativeloaf fifferences

44.loading process of rails server

45.file names in config firectory

46.difference between application.rb and environment.rb

47.asset pipelining

48.what are sweepers in rails.

49.directory structure of rails

50 difference between podt and get

51.rest architecture

52.how to create rake task

53.whst is action pack active support

54 . api integration in rails

55.calling controler methods in view

56.difference between gems and plugins

57.how to create custom actions in controller

58.perfirmance tuning of rails application

59.creating indexs in rails

60.ipmrove performance without creating indexes

61.send method usage

62.singleton methods

63.what is self

64.creating objects for classes inside module

65.validations

66.joins with ecamples

67.form\_for and form\_ differences

68.what are partials in rails

69.calling module methods in controller

70.creating a rails application without database.yml

71.requedt.xhr?

72.sorting array of object

#########$$$$$$$ java script

1) What is JavaScript?

Ans:JavaScript is a scripting language most often used for client-side web development.

2) What is the difference between JavaScript and Jscript?

Ans:Both JavaScript and Jscript are almost similar. JavaScript was developed by Netscape. Microsoft reverse engineered Javascript and called it JScript.

3) How do we add JavaScript onto a web page?

Ans:There are several way for adding JavaScript on a web page, but there are two ways which are commonly used by developers

If your script code is very short and only for single page, then following ways are the best:

a) You can place <script type="text/javascript"> tag inside the <head> element.

Code

<head>

<title>Page Title</title>

<script language="JavaScript" type="text/javascript">

var name = "Vikas Ahlawta"

alert(name);

</script>

</head>

b) If your script code is very large, then you can make a JavaScript file and add its path in the following way:

Code

<head>

<title>Page Title</title>

<script type="text/javascript" src="myjavascript.js"></script>

</head>

4) Is JavaScript case sensitive?

Ans:Yes!

A function getElementById is not the same as getElementbyID.

5) What are the types used in JavaScript?

Ans:String, Number, Boolean, Function, Object, Null, Undefined.

6) What are the boolean operators supported by JavaScript?

And Operator: &&

Or Operator: ||

Not Operator: !

7) What is the difference between “==” and “===”?

Ans:

“==” checks equality only,

“===” checks for equality as well as the type.

8) How to access the value of a textbox using JavaScript?

Ans:

<!DOCTYPE html>

<html>

<body>

Full name: <input type="text" id="txtFullName"

name="FirstName" value="Vikas Ahlawat">

</body>

</html>

There are following ways to access the value of the above textbox:

var name = document.getElementById('txtFullName').value;

alert(name);

or:we can use the old way:

document.forms[0].mybutton.

var name = document.forms[0].FirstName.value;

alert(name);

Note: This uses the "name" attribute of the element to locate it.

9) What are the ways of making comments in JavaScript?

Ans:

// is used for line comments

ex:- var x=10; //comment text

/\*

\*/ is used for block comments

ex:-

var x= 10; /\* this is

block comment example.\*/

10) How will you get the Checkbox status whether it is checked or not?

Ans:

var status = document.getElementById('checkbox1').checked;

alert(status);

will return true or false.

11) How to create arrays in JavaScript?

Ans:There are two ways to create array in JavaScript like other languages:

a) The first way to create array

Declare Array:

var names = new Array();

Add Elements in Array:-

names[0] = "Vikas";

names[1] = "Ashish";

names[2] = "Nikhil";

b) This is the second way:

var names = new Array("Vikas", "Ashish", "Nikhil");

12) If an array with name as "names" contain three elements, then how will you print the third element of this array?

Ans: Print third array element document.write(names[2]);

Note:- Array index starts with 0.

13) How do you submit a form using JavaScript?

Ans:Use document.forms[0].submit();

14) What does isNaN function do?

Ans: It returns true if the argument is not a number.

Code

document.write(isNaN("Hello")+ "<br>");

document.write(isNaN("2013/06/23")+ "<br>");

document.write(isNaN(123)+ "<br>");

The output will be:

true

true

false

15) What is the use of Math Object in JavaScript?

Ans: The math object provides you properties and methods for mathematical constants and functions.

Code

var x = Math.PI; // Returns PI

var y = Math.sqrt(16); // Returns the square root of 16

var z = Math.sin(90); Returns the sine of 90

16) What do you understand by this keyword in JavaScript?

Ans: In JavaScript the this is a context-pointer and not an object pointer. It gives you the top-most context that is placed on the stack. The following gives two different results (in the browser, where by-default the window object is the 0-level context):

var obj = { outerWidth : 20 };

function say() {

alert(this.outerWidth);

}

say();//will alert window.outerWidth

say.apply(obj);//will alert obj.outerWidth

17) What does "1"+2+4 evaluate to?

Ans: Since 1 is a string, everything is a string, so the result is 124.

18) What does 3+4+"7" evaluate to?

Ans: Since 3 and 4 are integers, this is number arithmetic, since 7 is a string, it is concatenation, so 77 is the result.

19) How do you change the style/class on any element using JavaScript?

Ans:

document.getElementById(“myText”).style.fontSize = “10";

-or-

document.getElementById(“myText”).className = “anyclass”;

20) Does JavaScript support foreach loop?

Ans: JavaScript 1.6(ECMAScript 5th Edition) support foreach loop,

See example here http://jsfiddle.net/gpDWk/

21) What looping structures are there in JavaScript?

Ans: for, while, do-while loops

22) What is an object in JavaScript, give an example?

Ans: An object is just a container for a collection of named values:

// Create the man object

Code

var man = new Object();

man.name = 'Vikas Ahlawat';

man.living = true;

man.age = 27;

23) How you will add function as a property in a JavaScript object? Give an example.

Ans:

Code

var man = new Object();

man.name = 'Vikas Ahlawat';

man.living = true;

man.age = 27;

man.getName = function() { return man.name;}

console.log(man.getName()); // Logs 'Vikas Ahlawat'.

24) What is the similarity between the 1st and 2nd statement?

1st:- var myString = new String('male'); // An object.

2nd:- var myStringLiteral = 'male'; // Primitive string value, not an object.

Ans: Both will call String() constructor function

You can confirm it by running the following statement:

console.log(myString.constructor, myStringLiteral.constructor);

25) What will be the output of the following statements?

var myString = 'Vikas' // Create a primitive string object.

var myStringCopy = myString; // Copy its value into a new variable.

var myString = null; // Manipulate the value

console.log(myString, myStringCopy);

Ans: // Logs 'null Vikas'

26) Consider the following statements and tell what would be the output of the logs statements?

var price1 = 10;

var price2 = 10;

var price3 = new Number('10'); // A complex numeric object because new was used.

console.log(price1 === price2);

console.log(price1 === price3);

Ans:

console.log(price1 === price2); // Logs true.

console.log(price1 === price3); /\* Logs false because price3

contains a complex number object and price 1

is a primitive value. \*/

27) What would be the output of the following statements?

var object1 = { same: 'same' };

var object2 = { same: 'same' };

console.log(object1 === object2);

Ans: // Logs false, JavaScript does not care that they are identical and of the same object type.

When comparing complex objects, they are equal only when they reference the same object (i.e., have the same address). Two variables containing identical objects are not equal to each other since they do not actually point at the same object.

28) What would be the output of the following statements?

var object1 = { same: 'same' };

var object2 = object1;

console.log(object1 === object2);

Ans: // Logs true

29) What is this?

var myArray = [[[]]];

Ans: Three dimensional array

30) Name any two JavaScript functions which are used to convert nonnumeric values into numbers?

Ans:

Number()

parseInt()

parseFloat()

Code

var n1 = Number(“Hello world!”); //NaN

var n2 = Number(“”); //0

var n3 = Number(“000010”); //10

var n4 = Number(true); //1

var n5 = Number(NaN); //NaN

31) Does JavaScript Support automatic type conversion, If yes give example.

Ans: Yes! Javascript support automatic type conversion. You should take advantage of it, It is most common way of type conversion used by Javascript developers.

var s = '5';

var a = s\*1;

var b = +s;

typeof(s); //"string"

typeof(a); //"number"

typeof(b); //"number"

Difference between window.onload and onDocumentReady?

The onload event does not fire until every last piece of the page is loaded, this includes css and images, which means there’s a huge delay before any code is executed.

That isnt what we want. We just want to wait until the DOM is loaded and is able to be manipulated. onDocumentReady allows the programmer to do that.

4. What is the difference between undefined value and null value?

undefined means a variable has been declared but has not yet been assigned a value. On the other hand, null is an assignment value. It can be assigned to a variable as a representation of no value.

Also, undefined and null are two distinct types: undefined is a type itself (undefined) while null is an object.

Unassigned variables are initialized by JavaScript with a default value of undefined. JavaScript never sets a value to null. That must be done programmatically.

6. What are Javascript closures?When would you use them?

Two one sentence summaries:

a closure is the local variables for a function – kept alive after the function has returned, or

\* a closure is a stack-frame which is not deallocated when the function returns.

A closure takes place when a function creates an environment that binds local variables to it in such a way that they are kept alive after the function has returned. A closure is a special kind of object that combines two things: a function, and any local variables that were in-scope at the time that the closure was created.

The following code returns a reference to a function:

function sayHello2(name) {

var text = ‘Hello ‘ + name; // local variable

var sayAlert = function() { alert(text); }

return sayAlert;

}

Closures reduce the need to pass state around the application. The inner function has access to the variables in the outer function so there is no need to store the information somewhere that the inner function can get it.

This is important when the inner function will be called after the outer function has exited. The most common example of this is when the inner function is being used to handle an event. In this case you get no control over the arguments that are passed to the function so using a closure to keep track of state can be very convenient.

7. What is unobtrusive javascript? How to add behavior to an element using javascript?

Unobtrusive Javascript refers to the argument that the purpose of markup is to describe a document’s structure, not its programmatic behavior and that combining the two negatively impacts a site’s maintainability. Inline event handlers are harder to use and maintain, when one needs to set several events on a single element or when one is using event delegation.

1

<input type="text" name="date" />

Say an input field with the name “date” had to be validated at runtime:

1

2

3

4

5

6

document.getElementsByName("date")[0].

addEventListener("change", validateDate, false);

function validateDate(){

// Do something when the content of the 'input' element with the name 'date' is changed.

}

Although there are some browser inconsistencies with the above code, so programmers usually go with a javascript library such as JQuery or YUI to attach behavior to an element like above.

8. What is Javascript namespacing? How and where is it used?

Using global variables in Javascript is evil and a bad practice. That being said, namespacing is used to bundle up all your functionality using a unique name. In JavaScript, a namespace is really just an object that you’ve attached all further methods, properties and objects. It promotes modularity and code reuse in the application.

10. What is the difference between innerHTML and append() in JavaScript?

InnerHTML is not standard, and its a String. The DOM is not, and although innerHTML is faster and less verbose, its better to use the DOM methods like appendChild(), firstChild.nodeValue, etc to alter innerHTML content.

#####$$$##$$$############

1.questions on guthub

2.what is single table inheritance and example

STI is basically the idea of using a single table to reflect multiple models that inherit from a base model, which itself inherits from ActiveRecord::Base.

In the database schema, sub-models are indicated by a single “type” column.

In Rails, adding a “type” column in a database migration is sufficient (after writing the models) to let Rails know that you’re planning to implement STI.

For example, I decided to use STI to convert what could have been one table/model per event type - a different table for football, basketball, hockey, festivals, concerts, etc. - into just two tables, sports and concerts.

Conveniently enough, this also mirrors Stubhub’s API, which categorizes events broadly into three categories - sports, concerts and theater (which I’ll aim to add at a later date).

Where STI is helpful is in the structure of the models that relate to the table. For my sports table, I have multiple models that each save data to the same table.

I have a class called “Sport” (I know, an awkward name, but I’m still getting used to the Rails pluralization conventions) that inherits from ActiveRecord::Base, and separate models called ProBasketball, ProFootball, Baseball, etc. that all inherit from the Sport class.

This enables each of the individual sports to take on the functionality of a Sport (and ActiveRecord by default), yet also provides me with sufficient flexibility to write lower-case sport-specific methods, constants and variables.

Each of these sports save to the “sports” table, with the sole differentiator being the “type” column, for which Rails uses the name of the Sport “sub-class” as the value (Rails just knows, don’t ask me how.).

Here is another example of Single Table Inheritance that you might find useful. You might have to support a hierarchy of users for you web application, such as User, Editor, and Administrator.

An Administrator might have full Create, Read, Update, and Delete (CRUD) access while a User might just have read access.

For this hierarchy of users you will need to create a Ruby class for each type and place it in the Rails app/models directory. Here is the base User class:

class User < ActiveRecord::Base

end

My Editor class:

class Editor < User

end

And the Administrator class:

class Administrator < Editor

end

Place each of these classes in their own Rails model file under the app/models directory.

To indicate to Ruby on Rails that the users table needs to support Single Table Inheritance you need to add a column named ‘type’ to the users table. Here is my users table definition:

CREATE TABLE users (

id INT NOT NULL AUTO\_INCREMENT,

user VARCHAR(15) NOT NULL UNIQUE,

pass VARCHAR(40) NOT NULL,

type VARCHAR(20) NOT NULL,

PRIMARY KEY (id)

);

In the column named type you should store the name of the class, the class type, that should be used for each user.

To mark an certain user as an admin set his type to ‘Administrator’. By setting a user’s type to ‘Administrator’ you are giving him full administrator privileges as defined in your Administrator model class.

And now, in your controller you can look up all users that are administrators by using the following bit of code:

allAdmins = Administrator.find(:all)

If you want to look up all known users you can do so by using the User class as in the following snippet of code:

allUsers = User.find(:all);

At this point, you would add additional responsibilities to editors and administrators by adding methods to those model classes.

3.what is polymorphic association and example

4.named scope

models/product.rb

class Product < ActiveRecord::Base

belongs\_to :category

named\_scope :cheap, :conditions => { :price => 0..5 }

named\_scope :recent, lambda { |\*args| {:conditions => ["released\_at > ?", (args.first || 2.weeks.ago)]} }

named\_scope :visible, :include => :category, :conditions => { 'categories.hidden' => false }

end

default\_scope(scope = {}) protected

Use this macro in your model to set a default scope for all operations on the model.

class Article < ActiveRecord::Base

default\_scope where(:published => true)

end

Article.all # => SELECT \* FROM articles WHERE published = true

The default\_scope is also applied while creating/building a record. It is not applied while updating a record.

Article.new.published # => true

Article.create.published # => true

You can also use default\_scope with a block, in order to have it lazily evaluated:

class Article < ActiveRecord::Base

default\_scope { where(:published\_at => Time.now - 1.week) }

end

(You can also pass any object which responds to call to the default\_scope macro, and it will be called when building the default scope.)

If you use multiple default\_scope declarations in your model then they will be merged together:

class Article < ActiveRecord::Base

default\_scope where(:published => true)

default\_scope where(:rating =>'G')

end

Article.all # => SELECT \* FROM articles WHERE published = true AND rating = 'G'

find and where

The difference is what they return when a record is found, or when it's not found. Consider the following examples:

>> User.create name: 'THE USER' # creates a user with id = 1

>> User.find(1) # returns the user

>> User.find\_by\_id(1) # returns the user

>> User.where(id: 1).first # returns the user

As you can see, an existing user can be fetched using any of the 3 methods. The big difference with using where is you can chain commands (of course, without calling first first.)

Let's have a look at when you try to find a record that isn't existing

>> User.find(2) # raises an exception

>> User.find\_by\_id(2) # nil

>> User.where(id: 2).first # nil

So here, it's obvious that when you use find to search for a record that isn't existing, you get an exception. That exception is ActiveRecord::RecordNotFound which renders a 404 on production environment.

custom validations in rails

class Invoice < ActiveRecord::Base

validate :expiration\_date\_cannot\_be\_in\_the\_past,

:discount\_cannot\_be\_greater\_than\_total\_value

def expiration\_date\_cannot\_be\_in\_the\_past

if expiration\_date.present? && expiration\_date < Date.today

errors.add(:expiration\_date, "can't be in the past")

end

end

def discount\_cannot\_be\_greater\_than\_total\_value

if discount > total\_value

errors.add(:discount, "can't be greater than total value")

end

end

end

5.difference between instance\_eval and class\_eval

Foo = Class.new

Foo.class\_eval do

def class\_bar

"class\_bar"

end

end

Foo.instance\_eval do

def instance\_bar

"instance\_bar"

end

end

Foo.class\_bar #=> undefined method ‘class\_bar’ for Foo:Class

Foo.new.class\_bar #=>"class\_bar"

Foo.instance\_bar #=>"instance\_bar"

Foo.new.instance\_bar #=> undefined method ‘instance\_bar’ for #<Foo:0x7dce8>

6.what blocks in rails

7.difference between rails 2 and rails3

8.difference between rails3 and rails4

9.fetchind second highest record from table

SELECT max(salary) FROM Employee WHERE salary NOT IN (SELECT max(salary) FROM Employee);

10.meta programming techniques in ruby

named scope , class\_eval, instance\_eval, send

11.method overloading and method overriding in ruby

12.different ways of calling a method in ruby

dot operator

send

instance\_eval

class\_eval

13.differece between module and call in ruby

we can create object for class and not for module.

14.mixins concept in ruby

one module can be mixed into ant number of classes.

16.observers in rsils

Observers are similar to callbacks, but with important differences. Whereas callbacks can pollute a model with code that isn’t directly related to its purpose, observers allow you to add the same functionality outside of a model.

imagine a User model where we want to send an email every time a new user is created. Because sending emails is not directly related to our model’s purpose, we could create an observer to contain this functionality.

class UserObserver < ActiveRecord::Observer

def after\_create(model)

# code to send confirmation email...

end

end

As with callback classes, the observer’s methods receive the observed model as a parameter.

17.caching in rails

page caching

action caching

fragment caching

sql cacging

18.filters and callbacks in rails

19.testing in rails mock test smoke test

20.how to upload files in ruby

21.updatind multiples models using single form?

using fields\_for

22.eager loading and lazy loading?

Eager Loading

One way to improve performance is to cut down on the number of SQL queries. You can do this through eager loading.

User.find(:all, :include => :friends)

Here you are firing only two queries :

1) One for all users.

2) One for all friends of users .

Lazy Loading :

When you have an object associated with many objects like a User has many Friends and you want to display a list as in Orkut you fire as many queries as there are friends, plus one for the object itself.

users = User.find(:all)

Then query for each user friend , like :

users.each do |user|

friend = Friend.find\_by\_user\_id(user.id)

end

Here

1) One query for all users.

2) N query for N no. of users friends

23.include and extend difference

include -includes module methods as instance methods

extend- includes module methods as self methods

24.proc and lamda difference

This is how I explain it… Ruby has Procs and Lambdas. Procs are created with Proc.new { }, lambdas are created with lambda {} and ->() {}.

In Ruby 1.8, proc {} creates lambda, and Ruby 1.9 it creates procs (don't ask).

Lambdas use method semantics when handling parameters, procs use assignment semantics when handling parameters.

This means lambdas, like methods, will raise an ArgumentError when called with fewer arguments than they were defined with. Procs will simply assign nil to variables for arguments that were not passed in.

Starting from Ruby 1.9, both procs and lambdas—like methods—support defining arguments with default values. To lambdas, such arguments are then optional at call time.

Lambdas act like anonymous functions w.r.t. the return statement (i.e. return will return from the lambda).

Procs are invisible to return, so a return statement will return from the enclosing function.

The technical way to say this is that Procs follow the Tennent Correspondence Principle w.r.t. return statements.

This allows you to wrap some code in a proc and immediately call the proc without changing the semantics of the code.

25. methid arguments &and \*

26.difference between private public and protrcted

27.implementing searching criteria in rails.

28.string and symbolsdigference

29. multiple methods with same name in rails

30.interpolation in rsils

31.deployment process how to deply a rails application

32.difference between habtm and has\_many through

33.explsin mvc architecture

34.number and cillection difference

35.render and redirect difference

36.delegate method usage

37.how to test a private method in rails

38.types testing in raild

39.how to test a controller sction in rails

40.what is irb

41.difference bet gemfile and gemfile.lock

42.what is bundler

43.load require autoload relativeloaf fifferences

44.loading process of rails server

45.file names in config firectory

46.difference between application.rb and environment.rb

47.asset pipelining

48.what are sweepers in rails.

49.directory structure of rails

50 difference between podt and get

51.rest architecture

52.how to create rake task

53.whst is action pack active support

54 . api integration in rails

55.calling controler methods in view

56.difference between gems and plugins

57.how to create custom actions in controller

58.perfirmance tuning of rails application

59.creating indexs in rails

60.ipmrove performance without creating indexes

61.send method usage

62.singleton methods

63.what is self

64.creating objects for classes inside module

65.validations

66.joins with ecamples

67.form\_for and form\_ differences

68.what are partials in rails

69.calling module methods in controller

70.creating a rails application without database.yml

71.requedt.xhr?

72.sorting array of object

Different ways of implementing many to many relationships in rails ?

There's basically two ways: has\_and\_belongs\_to\_many (habtm) and has\_many with a :through option that points to another association.

Both require join tables; the latter is what we call a join model, because you typically add more information to the join.

For example, consider an application with a User model who bookmarks Sites. One way would be to implement it as a habtm relationship

class User <ActiveRecord::Base  
  has\_and\_belongs\_to\_many :sites  
end  
class Site <ActiveRecord::Base  
  has\_and\_belongs\_to\_many :users  
end  
  
user.sites<<Site.find(...)

This modeling will also require creating the sites\_users table, which necessarily will lack a primary key.

The problem with this is you're likely to want to store additional information on it, so you might as well go with a join model, in this case Bookmark:

class User <ActiveRecord::Base  
  has\_many :bookmarks  
  has\_many :sites, :through => :bookmarks  
end  
class Site <ActiveRecord::Base  
  has\_many :bookmarks  
  has\_many :users, :through => :bookmarks  
  #edit: adding validation for requiring at least one bookmark  
  validate\_before\_create :at\_least\_one\_bookmark  
  private  
  def at\_least\_one\_bookmark  
    errors.add\_to\_base("requires at least one bookmark") unless bookmarks.count> 0  
  end  
end  
class Bookmark <ActiveRecord::Base  
  belongs\_to :user  
  belongs\_to :site  
end  
  
user.bookmarks.create(:site =>Site.find(...) )

The more common pattern is the join model approach for its versatility and better modelling, though habtms are still used somewhat.

They're just so two-dimensional that you really need to examine what you're doing and make sure there isn't some richer behavior that needs to be modelled as well.

## Basic many-to-many Associations in Rails

### Many-to-many relationships

There are two ways to handle many-to-many relationships in Ruby on Rails, and this article will cover both.

### has\_and\_belongs\_to\_many

The simplest approach is if you don’t need to store any information about the relationship itself. You just want to know what items are in each category, and what categories each item belongs to. This is called “has\_and\_belongs\_to\_many”. We use has\_and\_belongs\_to\_many associations in our models, and create a join table in our database. Here are your models:

|  |  |
| --- | --- |
| 1 | # app/models/category.rb |
| 2 | classCategory <ActiveRecord::Base | |

|  |  |  |
| --- | --- | --- |
| 3 | has\_and\_belongs\_to\_many :items | |
| 4 | end |

|  |  |
| --- | --- |
| 5 |  |
| 6 | # app/models/item.rb | |

|  |  |
| --- | --- |
| 7 | classItem <ActiveRecord::Base |
| 8 | has\_and\_belongs\_to\_many :categories | |

|  |  |
| --- | --- |
| 9 | end |

Next, let’s create the join table by generating a new migration. From the command line:

|  |  |
| --- | --- |
| 1 | script/generate migration AddCategoriesItemsJoinTable |

Now we’ll edit the migration file it creates:

|  |  |  |
| --- | --- | --- |
| 01 | classAddCategoriesItemsJoinTable<ActiveRecord::Migration | |
| 02 | defself.up |

|  |  |  |
| --- | --- | --- |
| 03 | create\_table :categories\_items, :id=> falsedo|t| | |
| 04 | t.integer :category\_id |

|  |  |  |
| --- | --- | --- |
| 05 | t.integer :item\_id | |
| 06 | end |

|  |  |  |
| --- | --- | --- |
| 07 | end | |
| 08 |  |

|  |  |
| --- | --- |
| 09 | defself.down |
| 10 | drop\_table :categories\_items | |

|  |  |  |
| --- | --- | --- |
| 11 | end | |
| 12 | end |

Notice the :id => false, which keeps the migration from generating a primary key. The name of the table is a combination of the two table names we’re joining, in alphabetical order. This is how Rails knows how to find the join table automatically.

### has\_many :through

The other way to setup a many-to-many relationship between objects is used if you do, or think you will, need to track info on the relationship itself.

When was item X added to category Y? That’s info you can’t store in the category or item tables, because it’s info about the *relationship*.

In Rails, this is called a has\_many :through association, and it’s really just as easy as the first way.

First, we’re going to create a new model, that defines the relationship between items and categories. For back of a better name, let’s call it a Categorization. Setup your models like this:

|  |  |
| --- | --- |
| 01 | # app/models/category.rb |
| 02 | classCategory <ActiveRecord::Base | |

|  |  |
| --- | --- |
| 03 | has\_many :categorizations |
| 04 | has\_many :items, :through=> :categorizations | |

|  |  |  |
| --- | --- | --- |
| 05 | end | |
| 06 |  |

|  |  |
| --- | --- |
| 07 | # app/models/item.rb |
| 08 | classItem <ActiveRecord::Base | |

|  |  |
| --- | --- |
| 09 | has\_many :categorizations |
| 10 | has\_many :categories, :through=> :categorizations | |

|  |  |  |
| --- | --- | --- |
| 11 | end | |
| 12 |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 13 | # app/models/categorization.rb |  | |  | |  |  | |  | |  |  | |  | |  |  | |  | |  |  | |  | |
| 14 | classCategorization <ActiveRecord::Base | |  | |  |  | |  | |  |  | |  | |  |  | |  | |  |  | |  | |  | |

|  |  |  |
| --- | --- | --- |
| 15 | belongs\_to :category | |
| 16 | belongs\_to :item |

|  |  |
| --- | --- |
| 17 | end |

We’re connecting both original models to :categorizations, and then connecting the them to each other *via* the intermediary Categorization model. Now, instead of a join table whose only function is connecting the others, we add a full-fledged table to manage our new model:

|  |  |  |
| --- | --- | --- |
| 01 | classCreateCategorizations<ActiveRecord::Migration | |
| 02 | defself.up |

|  |  |  |
| --- | --- | --- |
| 03 | create\_table :categorizationsdo|t| | |
| 04 | t.integer :category\_id |

|  |  |  |
| --- | --- | --- |
| 05 | t.integer :item\_id | |
| 06 |  |

|  |  |  |
| --- | --- | --- |
| 07 | t.timestamps | |
| 08 | end |

|  |  |
| --- | --- |
| 09 | end |
| 10 |  | |

|  |  |
| --- | --- |
| 11 | defself.down |
| 12 | drop\_table :categorizations | |

|  |  |  |
| --- | --- | --- |
| 13 | end | |
| 14 | end |

We still have the two foreign key integer columns, but we’ve removed :id => false so this table will have an id column of its own. We also added timestamps, so we’ll be able to tell when an item was added to a specific category. I also created a migration that removes the old categories\_items table, but it’s not shown here.

### Which is Better?

The simpler has\_and\_belongs\_to\_many approach has a small advantage when you \*know\* you’re not going to need to track info about the relationship itself.

If this is the case, there’s a very *slight* performance gain because you’re not loading an extra model class at runtime.

More often than not, however, you’re going to eventually want to track relationship-specific data.

We used the example of tracking when a relationship was created.

Another would be if you want to track, over time, how many times a visitor clicks on an item under each category.

That counter needs to be stored in the Categorization model, and that’s a reason not to use the simpler has\_and\_belongs\_to\_many approach.

I’ve created an example application ([get it here](http://github.com/bellmyer/many_to_many_examples)) with tags for each version – has\_and\_belongs\_to\_many, and has\_many :through.

============================================================================

What are model level callbacks ?

1. before\_validation
2. before\_create
3. before\_destroy

what is the difference between member and collection ?

A member route will require an ID, because it acts on a member.

A collection route doesn't because it acts on a collection of objects.

Preview is an example of a member route, because it acts on (and displays) a single object.

Search is an example of a collection route, because it acts on (and displays) a collection of objects.

resources :photos do  
  member do  
    get :preview  
  end  
end  
  
# vs  
  
resources :photos do  
  collection do  
    get :search  
  end  
end

What is deffirence between http and https ?

what is the difference between http and https  
and where would i use it and how do i configure it in tomcat

1. HTTP is Hyper Text Transport Protocol and is transmitted over the wire via PORT 80(TCP).
2. You normally use HTTP when you are browsing the web, its not secure, so someone can eavesdrop on the conversation between your computer and the web server.
3. HTTPS (Hypertext Transfer Protocol over Secure Socket Layer, or HTTP over SSL) is a Web protocol developed by Netscape and built into its browser that encrypts and decrypts user page requests as well as the pages that are returned by the Web server
4. HTTPS is really just the use of Netscape's Secure Socket Layer (SSL) as a sublayer under its regular HTTP application layering.
5. (HTTPS uses port 443 instead of HTTP port 80 in its interactions with the lower layer, TCP/IP.)
6. SSL uses a 40-bit key size for the RC4 stream encryption algorithm,new-age browsers use 128-bit key size which is more secure than the former, it is considered an adequate degree of encryption for commercial exchange
7. HTTPS is normally used in login pages, shopping/commercial sites.
8. Although it may be encrypted does not mean its safe, there are tools out there to decrypt the information being sent over the wire, although its more difficult to do so.

Why Ruby on Rails?

There are lots of advantages using ruby on rails

1. DRY Principal

2. Convention over Configuration

3. Gems and Plugins

1. Plugins are just libraries loaded from a specific directory, gems are loaded via Bundler or RubyGems directly.
2. Where this really makes a differences is maintenance and management.
3. What happens when you want the latest and greatest authlogicplugin, well you need to update the files in your directory.
4. That doesn't sound so bad when it's one plugin, but what about something that constantly updates? There was/is an existing system for code packaging and distribution (RubyGems), which lends itself to managing such things.
5. Consider the authlogic example again, what happens if the new version requires some other dependency now? With RubyGems the gem file explicitly defines that relationship, the plugin system does not and such a definition would've been redundant

4. Scaffolding

**Scaffolding** is a [meta-programming](http://en.wikipedia.org/wiki/Meta-programming) method of building [database](http://en.wikipedia.org/wiki/Database)-backend [software applications](http://en.wikipedia.org/wiki/Software_application" \o "Software application)

. It is a technique supported by some [model-view-controller](http://en.wikipedia.org/wiki/Model-view-controller)[frameworks](http://en.wikipedia.org/wiki/Software_framework), in which the programmer may write a specification that describes how the application database may be used

5. Pure OOP Concept

6. Rest Support

HTTP and REST

Although some people confuse them, HTTP and REST are not one and the same,and we keep to that distinction.

We have one section dealing with HTTP and covering the basics: URI, open-uri, Net::HTTP, Mongrel.

The REST part expands on it to deal with resources and representations, and does so using Rails 2.0 and ActiveResource

7. Rack support

1. Supporting Rake version: 0.8.3
2. This package contains Rake, a simple ruby build program with capabilities similar to make.

Rake has the following features:

1. Rakefiles (rake‘s version of Makefiles) are completely defined in standard Ruby syntax. No XML files to edit. No quirky Makefile syntax to worry about (is that a tab or a space?)
2. Users can specify tasks with prerequisites.
3. Rake supports rule patterns to synthesize implicit tasks.
4. Flexible FileLists that act like arrays but know about manipulating file names and paths.
5. A library of prepackaged tasks to make building rakefiles easier.

8. Action Mailer

1. Action Mailer allows you to send emails from your application using a mailer model and views.
2. So, in Rails, emails are used by creating mailers that inherit from ActionMailer::Base and live in app/mailers.
3. Those mailers have associated views that appear alongside controller views in app/views.

9. Rpc support

1. Ruby RPC is an alternative to DRb.
2. It provides a similar service, but in a way that is slightly more compatible with my way of thinking.
3. It dispenses with the URL syntax and some of the complexity of DRb and adds automatic distributed reference counting garbage collection.
4. This prevents the problem seen in DRb where objects disappear when there are no local references to them.
5. The interface is also designed to be simple and transparent, with a minimum of programmer work needed to use it.
6. Ruby RPC was developed and is maintained by Brian Ollenberger. If you need to contact me, see the contact link above.

Features

Ruby RPC has the following features and characteristics:

•Transparent interface.

•Allows serving a single object on a single TCP port.

•Allows other arbitrary sockets, such as pipes, UNIX sockets or SSL sockets.

•Provides distributed reference counting garbage collection.

•Fully supports calling remote methods with blocks, totally transparently.

•Automatically optimizes away many unnecessary round trips.

•Works with another project, of mine, Ruby ODB, to provide remote access to an object database

10. RexmlSupport ::

1. Parse XML using Ruby
2. The typical way to parse XML in Ruby is to use REXML, which comes as part of the standard library.

2. Explain about the programming language ruby?

Ruby is the brain child of a Japanese programmer Matz. He created Ruby. It is a cross platform object oriented language. With legacy code it gives you the power of administration and organization tasks. Being open source, it did go into great lengths of development.

3. Explain about ruby names?

Classes, variables, methods, constants and modules can be referred by ruby names

4. What is the Difference between Symbol and String?

Ans: Symbol are same like string but both behaviors is different based on object\_id, memory and process time (cpu time) Strings are mutable , Symbols are immutable.

Mutable objects can be changed aftern assignmentwhile immutable objects can only be overwritten. For example

p"string object jak".object\_id #=> 22956070

p"string object jak".object\_id #=> 22956030

p"string object jak".object\_id #=> 22956090

p :symbol\_object\_jak.object\_id #=> 247378

p :symbol\_object\_jak.object\_id #=> 247378

p :symbol\_object\_jak.object\_id #=> 247378

p" string object jak".to\_sym.object\_id #=> 247518

p" string object jak".to\_sym.object\_id #=> 247518

p" string object jak".to\_sym.object\_id #=> 247518

p :symbol\_object\_jak.to\_s.object\_id #=> 22704460

p :symbol\_object\_jak.to\_s.object\_id #=> 22687010

p :symbol\_object\_jak.to\_s.object\_id #=> 21141310

And also it will differ by process time

Testing two symbol values for equality (or non-equality) is faster than testing two string values for equality,

Note : Each unique string value has an associated symbol

5. What is Session and Cookies?

Ans: Session: are used to store user information on the server side.

cookies: are used to store information on the browser side or we can say client side

Session : say session[:user] = “arunkumar” it remains when the browser is not closed

6. What is request.xhr?

Ans: A request.xhr tells the controller that the new Ajax request has come, It always return Boolean values (TRUE or FALSE)

7. What is MVC? and how it Works?

Ans: MVC tends for Model-View-Controller, used by many languages like PHP, Perl, Python etc. The flow goes like this: Request first comes to the controller, controller finds and appropriate view and interacts with model, model interacts with your database and send the response to controller then controller based on the response give the output parameter to view, for Example your url is something like this:

http://localhost:3000/users/new

here users is your controller and new is your method, there must be a file in your views/users folder named new.html.erb, so once the submit button is pressed, User model or whatever defined in the rhtmlform\_for syntax, will be called and values will be stored into the database.

8. What things we can define in the model?

Ans: There are lot of things you can define in models few are:

1. Validations (like validates\_presence\_of, numeracility\_of, format\_of etc.)

2. Relationships(like has\_one, has\_many, HABTM etc.)

3. Callbacks(like before\_save, after\_save, before\_create etc.)

4. Suppose you installed a plugin say validation\_group, So you can also define validation\_group settings in your model

5. ROR Queries in Sql

6. Active record Associations Relationship

2 The Types of Associations

In Rails, an association is a connection between two Active Record models. Associations are implemented using macro-style calls, so that you can declaratively add features to your models. For example, by declaring that one model belongs\_to another, you instruct Rails to maintain Primary Key–Foreign Key information between instances of the two models, and you also get a number of utility methods added to your model. Rails supports six types of associations:

belongs\_to

has\_one

has\_many

has\_many :through

has\_one :through

has\_and\_belongs\_to\_many

In the remainder of this guide, you’ll learn how to declare and use the various forms of associations. But first, a quick introduction to the situations where each association type is appropriate.

9. What is ORM in Rails?

Ans: ORM tends for Object-Relationship-Model, it means that your Classes are mapped to table in the database, and Objects are directly mapped to the rows in the table.

10. How many Types of Associations Relationships does a Model has?

Ans: When you have more than one model in your rails application, you would need to create connection between those models. You can do this via associations. Active Record supports three types of associations:

one-to-one : A one-to-one relationship exists when one item has exactly one of another item. For example, a person has exactly one birthday or a dog has exactly one owner.

one-to-many : A one-to-many relationship exists when a single object can be a member of many other objects. For instance, one subject can have many books.

many-to-many : A many-to-many relationship exists when the first object is related to one or more of a second object, and the second object is related to one or many of the first object.

You indicate these associations by adding declarations to your models: has\_one, has\_many, belongs\_to, and has\_and\_belongs\_to\_many.

18. How many types of callbacks available in ROR?

Ans:

(-) save

(-) valid

(1) before\_validation

(2) before\_validation\_on\_create

(-) validate

(-) validate\_on\_create

(3) after\_validation

(4) after\_validation\_on\_create

(5) before\_save

(6) before\_create

(-) create

(7) after\_create

(8) after\_save

19. What can Rails Migration do?

20. What is the naming conventions for methods that return a boolean result?

Ans:

Methods that return a boolean result are typically named with a ending question mark.

For example:def active? return true #just always returning true end

21. How do the following methods differ: @my\_string.strip and @my\_string.strip! ?

Ans: The strip! method modifies the variable directly. Calling strip (without the !) returns a copy of the variable with the modifications, the original variable is not altered.

22. What's the difference in scope for these two variables: @name and @@name?

Ans: @name is an instance variable and @@name is a class variable

23. What is the log that has to seen to check for an error in ruby rails?

Ans: Rails will report errors from Apache in log/apache.log and errors from the Ruby code in log/development.log

24. What is the use of global variable $ in Ruby?

To write a global variable you start the variable with a $ sign which should be followed by a name character. Ruby defines a number of global variables which also include other punctuation characters such as $\_ and $-k.

25. Where does the start\_tabnavgetinformations for tabs rendering in ruby rail?

Ans: The main Symbol let the start\_tabnav method know to look for a special MainTabnav class where all the magic happens

26. What is the Install rail package?

Ans: There are several packages that you can download and install. The prebuilt Rails installer called Install rail which currently is only for Windows

28. What is the use of super in ruby rails?

Ans: Ruby uses the super keyword to call the superclass (Parent class) implementation of the current method

29. What is the difference between nil and false in ruby?

Ans: False is a booleandatatype, Nil is not a data type it have object\_id 4

30. How is class methods defined in Ruby?

Ans: A:

defself.methodname

ordefclassname.methodname

32. What are the priority of operators available in Ruby ?

Ans: Something that used in an expression to manipulate objects such as + (plus), - (minus), \* (multiply), and / (divide). You can also use operators to do comparisons,such as with <, >, and &&. The priority is based on "BODMAS"

33. What are the looping structures available in Ruby?

Ans: for..in

untill..end

while..end

do..end

Note: You can also use each to iterate a array as loop not exactly like loop

34. What are the object-oriented programming features supported by Ruby and how multiple inheritance supported in ?

Ans: Classes,Objects,Inheritance,Singletonmethods,polymorphism(accomplished by over riding and overloading) are some oo concepts supported by ruby. Multiple inheritance supported using Mixin concept.

In this example, test1 and test2 belong to same class, but test2 has been given a redefined size method and so they behave differently.

A method given only to a single object is called a singleton method.

Singleton methods are often used for elements of a graphic user interface (GUI), where different actions need to be taken when different buttons are pressed.

Singleton methods are not unique to ruby, as they appear in CLOS, Dylan, etc. Also, some languages, for example, Self and NewtonScript, have singleton methods only.

These are sometimes called prototype-based languages.

35. What is the scope of a local variable in Ruby and define it scope ?

Ans: A new scope for a local variable is introduced in the toplevel, a class (module) definition, a method defintion.

In a procedure block a new scope is introduced but you can access to a local variable outside the block.

The scope in a block is special because a local variable should be localized in Thread and Proc objects.

36. How is an enumerator iterator handled in Ruby?

Ans: Iterator is handled using keyword 'each' in ruby.

For example

number=[1,2,3]

then we can use iterator as

number.each do |i|

putsi

end

Above prints the values of an array $no which is accomplished using iterator.

37. How is visibility of methods changed in Ruby (Encapsulation)?

Ans: By applying the access modifier : Public , Private and Protected access Modifier

38. What is the use of load,require, auto\_load,require\_relative in Ruby?

Ans:

A require method that loads and processes the Ruby code from a separate file, including whatever classes, modules, methods, and constants are in that file into the current scope. load is similar, but rather than performing the inclusion operation once, it reprocesses the code every time load is called

auto\_load - Whenever the interpreter call the method that time only it will initiate the method in hat file.

require\_relative - It it to load local folder files.

34.bundlers

Bundler manages an **application's dependencies** through its entire life across many machines systematically and repeatably.

1. Explain choose\_weighted method with example

2. What is GIL in ruby ?

A Global Interpreter Lock (GIL) is a mutual exclusion lock held by a programming language interpreter thread to avoid sharing code that is not thread-safe with other threads

3. Is variable is a object ?

Ans : Variable is not an object

4. List of protocols supported by ruby ?

5. Explain Virtual attribute ?

6. How to validate and modify attribute value ?

7. How to serialize data with YAML ?

2.2 What is is Agile Web Development with Rails?

2.3 What is a Model ?

The model is responsible for maintaining the state of the application. model

Sometimes this state is transient, lasting for just a couple of interactions

with the user. Sometimes the state is permanent, and will be stored outside

the application, often in a database.

A model is more than just data; it enforces all the business rules that apply

to that data.

2.4 What is a View?

The view is responsible for generating a user interface, normally based view

on data in the model..5

What are Controllers?

Controllers orchestrate the application. Controllers receive incoming events Controllers

(normally user input), interact with the model, and display an appropriate

view to the user.

6 Explain the process from request to response in rails MVC?

IN Rails application, incoming requests are first sent to a router, which

works out where in the application the request should be sent, and how

the request itself should be parsed. Ultimately, this phase identifies a

particular method (called an action in Rails parlance) somewhere in the action

controller code. The action might look at data in the request itself, it

might interact with the model, and it might cause other actions to be

invoked. Eventually the action prepares information for the view which

renders something to the user

2.8 What is Active Record ?

Active Record is the ORM layer supplied with Rails. It closely follows the

standard ORM model: tables map to classes, rows to objects, and columns

to object attributes.

2.9 What is script directory ?

The scripts directory holds programs that are useful for developers. Run

any of these scripts with no arguments to get usage information.

benchmarker

Get performance benchmarks on one or more methods in your application.

breakpointer

A client that lets you interact with running Rails applications. We

talk about this starting on page 189.

console

Allows you to use irb to interact with your Rails application methods.irb

destroy

Removes autogenerated files created by generate.

generate

A code generator. Out of the box, it will create controllers, mailers,models, scaffolds, and web services

profiler

Creates a runtime-profile summary of a chunk of code from your

application.

runner

Executes a method in your application outside the context of the web.

You could use this to invoke cache expiry methods from a cron job,

or handle incoming e-mail.

server

A WEBrick-based server that will run your application

What Is Ruby on Rails - Ruby on Rails is a web application framework written in Ruby, a dynamically typed programming language similar to Python, Smalltalk, and Perl.

Ruby on Rails extensive Wikipedia entry. Ruby on Rails, often called RoR, or just Rails, is an open source web application framework written in Ruby that closely follows the Model-View-Controller (MVC) architecture

. It strives for simplicity and allowing real-world applications to be developed in less code than other frameworks and with a minimum of configuration

. The Ruby programming language allows for extensive metaprogramming, which Rails makes much use of. This results in a syntax that many of its users find to be very readable.

Rails is primarily distributed through RubyGems, which is the official packaging format and distribution channel for Ruby libraries and applications.

Exploring Ruby on Rails - from Linux Journal. It seemed that every blog I read either was proclaiming Rails as the new juggernaut of Web frameworks or was damning it as the scourge of developers everywhere. Now, I generally assume anything that’s simultaneously causing so much adoration, protest and reflection must have something going for it, and rumors that Dave Thomas was putting together a book on RoR only fueled my motivation to find out all that I could as fast as I could. So I installed Rails, raced through a few tutorials, started reading the source and called Doug to get the lowdown straight from the horse’s mouth.

Get On Track with Ruby on Rails - Ruby? Rails? Ajax? Is this cluster of confusing buzzwords swarming around your head like a mob of angry bees? Then fear not, gentle padawan — Jedi help is at hand! For I, too, was just like you once: confused, jumping at shadows, sinking into the depths of yet another custom Python web framework while softly weeping behind a stack of dog-eared PHP books. But not anymore. Today I build my applications in Ruby on Rails.

Ruby on Rails: An Interview with David Heinemeier Hansson - Few can have missed the rise of the programming world’s latest star platformÃ¢â‚¬â€Ruby on Rails. Rails’ creator, David Heinemeier Hansson, already wowed the crowds at this year’s OSCON and is set to keynote the European O’Reilly Open Source Convention in Amsterdam this October.

A good analysis of pros and cons of Ruby on Rails - I recently finished using Ruby on Rails to write a simple bug tracking application. I thought I’d take this new RAD environment for a spin, having heard all the hype.

Getting Your Feet Wet With Ruby on Rails - If you’re like me, you’re reading this on your bright-red custom-built laptop in a soothing rosemary-scented bubble bath, and you’re wondering, “Why do I want another interpreted programming language? I can find my way around Perl and PHP and maybe a little Python. And Unix shell scripting. I feel fine. Why do people keep talking about Ruby?”

Fast-track your Web apps with Ruby on Rails - Ruby on Rails is a recent entry into the world of Web application development that is rapidly gaining mindshare, even while still in beta versions. Rails succeeds by automating the creation of the most common types of Web applications while not straightjacketing you if you want to add custom or atypical requirements. Moreover, compared to many Free Software libraries that perform individual aspects of a Web application, Rails contains a nicely integrated set of tools for all aspects.

Ruby on Rails HOWTOs - off the official wiki

Ruby On Rails manuals - linked on the official site

Ruby on Rails playing with other technologies

Installing Ruby on Rails with Lighttpd and MySQL on Fedora Core 4 - This how-to is aimed at getting you started with Ruby on Rails installed on the Fedora Core 4 Linux distribution using the Lighttpd web server and mySQL database server.

Ajax on Rails - In a few short months, Ajax has moved from an obscure and rarely used technology to the hottest thing since sliced bread. This article introduces the incredibly easy-to-use Ajax support that is part of the Ruby on Rails web application framework. This is not a step-by-step tutorial, and I assume that you know a little bit about how to organize and construct a Rails web application.

PHPEquivalents - a tutorial for those migrating from PHP applications

Ruby off the Rails - Ruby on Rails is just one facet of what makes Ruby great, just like EJB is only part of the Java enterprise platform. Andrew Glover digs beneath the hype for a look at what Java developers can do with Ruby, all by itself.

Speed up Ruby-on-Rails with memcached - So, even though I donÃ¢â‚¬â„¢t have a huge amount of traffic, I still have dynamic sites, and IÃ¢â‚¬â„¢m always looking at ways to speed up my Typo blog (this site). So, using memcached, you can get a big performance boost in databases calls, which sold me on giving it a go. I read two posts today, one about howto set this up in Freebsd, along with howto modify the source code for a boost over the default setting, and how to make Ruby-on-Rails take advantage of it. Below are steps compiled from both sites, and used on my FreeBSD 6.0 server, but most of the steps should work as well in Linux. Read more for the steps.

Ruby on Rails on Oracle: A Simple Tutorial - Rails is a capable Web application platform and has, in less than two years, gained significant traction among J2EE and PHP programmers. The attraction of both J2EE and PHP programmers makes a lot of sense when you look at the strengths of Rails. For one thing, it uses a strict model-view-controller architecture that any self-respecting design-patterns wonk would admireÃ¢â‚¬â€this explains its attraction to J2EE developers. Second, itÃ¢â‚¬â„¢s easy to build basic systems with RailsÃ¢â‚¬â€which is attractive to PHP developers.

Using Ruby on Rails on Dreamhost - DreamHost is a popular Web hosting provider with good plans, and hence is popular among many Webmasters who are just starting up.

1.Difference between html.erb&rhtml

It's just a change of philosophy between Rails 1 and Rails 2.Before Rails 2, we had file.rhtml, file.rxml and file.rjs.In Rails 2.0, that changed to file.content\_type.template\_engine. So with file.html.erb,the content type is html and the template engine is ERB. rxml is now xml.builder and rjs should now (mostly) be js.rjs.

2.What is dynamic database

3.Difference between dynamic hasfinder and hasfinder

Most common operations in many applications is to simply query based on one or two columns, Rails has an easy and effective way to do these queries

without having to resort to the conditions parameter of find.Dynamic finder methods begin with find\_by\_name or find\_all\_by\_name, indicating whether you want a single value or array of results returned. The semantics are similar to calling find with the :first versus the :all option.

>City.find\_by\_name("Hackensack")

=> # "Hackensack", "latitude" =>"40.8858330000", "id" =>"15942", "longitude" =>"-74.0438890000","state" =>"NJ" }>

>>City.find\_all\_by\_name("Atlanta").collect(&:state)

=> ["GA", "MI", "TX"]

It's also possible to use multiple attributes in the same find by separating them with "and", so you get finders like Person.find\_by\_user\_name\_and\_password or

evenPayment.find\_by\_purchaser\_and\_state\_and\_country.

Has finder or named\_scope is an extension to ActiveRecord that makes it easy to create custom finder and count methods on your ActiveRecord models.

10.What is meta programming

11.What is Rspec

12.Tell briefly about RESTFUL Architecture

13.Difference between map.resources and map.resource

14.What is a block

15.Define Lambda & Proc

16.Difference between Lambda & Proc

17.Three levels of protection in ruby

18.Difference between protected method and private method

19.What is attr\_accessor, attr\_protected

20.How a website is working in Rails application

21.What is sessions and cookies

22.How sessions are storing

23.What is gem and difference between gem and plugin

24.What is TDD

25.What are callbacks

26.Why ruby is called as dynamic language

27.What is namespace

28.What is left join and outer join

29.How many engines mysql is having

30.What are called modules and what is the use of lib folder

Modules are a way of grouping together methods, classes, and constants. Modules give you two major benefits.

1. Modules provide a namespace and prevent name clashes.

2. Modules support the mixin facility.

Lib directory holds application code that doesn’t fit neatly into a model,

view, or controller.

The lib directory is also a good place to put code that’s shared among models,views, or controllers

.If the file is in the lib directoryitself, you require it directly by name.

31.How to provide grant permission for mysql

GRANT ALL PRIVILEGES ON my\_database.\* TO 'my\_user'@'localhost' IDENTIFIED BY 'my\_password' WITH GRANT OPTION;

32.Difference between single quote and double quote in ruby

Single-quoted strings are treated as literals;

double-quoted strings are interpolated.Single quotes will print the value whereas double quotes will execute it.

For example

>> r = 42

=> 42

>>"#{r}"

=>"42"

>>'#{r}'

=>"\#{r}"

33.Why ruby?

They are interpreted and do not need a compile phase, which increases development speed tremendously. After editing your program you can see the results of your changes immediately.

• Enterprise software is about munging data. Dynamic languages are designed to handle data, and include high-level data types such as hashes.

• Memory management is dealt with by the language. This is a great advantage over languages such as C++ where you have to specify the length of each string you read from a database. Dynamic languages prevent waste and result in more concise, more robust,and more secure software.

• Software written in dynamic languages is installed as source code,so you always know exactly which version is currently running on your production system. Gone are the days when you had to guess if a certain binary executable is the right one.

Q2. What is the Difference between Static and Dynamic Scaffolding?

Sol: The Syntax of Static Scaffold is like this:

ruby script/generate scaffold User Comment

Where Comment is the model and User is your controller, So all n all static scaffold takes 2 parameter i.e your controller name and model name, whereas in dynamic scaffolding you have to define controller and model one by one.

Q11. What is the difference between rails version 2.2.2 with the older ones?

Q12. Difference between render and redirect?

Q13. How to use sqldb or mysqldb.without defining it in the database.yml

Sol. http://stuff.lilleaas.net/active\_record\_anywhere

Q14. What are helpers and how to use helpers in ROR?

Sol. Helpers (“view helpers”) are modules that provide methods which are automatically usable in your view.

They provide shortcuts to commonly used display code and a way for you to keep the programming out of your views.

The purpose of a helper is to simplify the view.

It’s best if the view file (RHTML/RXML) is short and sweet, so you can see the structure of the output.

Q15. What is Active Record?

Sol. Active Record are like Object Relational Mapping(ORM), where classes are mapped to table and objects are mapped to colums in the table

Q16. Ruby Supports Single Inheritence/Multiple Inheritence or Both?

Sol. Ruby Supports only Single Inheritnece

Q18. Suppose in one of my method I am updating the attributes of table, in my model I have defined after\_create do X, and after\_save do Y. Which method will be called?

Q19. How to use two database into a Single Application?

Sol. http://magicmodels.rubyforge.org/magic\_multi\_connections/, According to this link :ActiveRecord models are allowed one connection to a database at a time, per class.

Ruby on Rails sets up the default connection based on your database.yml configuration to automatically select development, test or production.

But, what if you want to access two or more databases – have 2+ connections open – at the same time.

ActiveRecord requires that you subclass ActiveRecord::Base. That prevents you doing migrations from one database to another.

It prevents you using one set of model classes on two or more databases with the same schema.

Magic Multi-Connections allows you to write your models once, and use them for multiple Rails databases at the same time. How?

Using magical namespacing.

To do this :

[A] sudo gem install magic\_multi\_connections

[B] require ‘magic\_multi\_connections’

Add the following to the bottom of your environment.rb file

You can also find examples on this link : http://magicmodels.rubyforge.org/magic\_multi\_connections/

“Updated Set of Questions for Rails 3”

Q22. What is the Newest approach for find(:all) in Rails 3?

Sol: Model.where(:activated => true)

Q23. What is Gemfile and Gemfile.lock?

Q1. What is the Notation used for denoting class variables in Ruby?

Q2. What is the use of Destructive Method?

Q3. What is the use of load and require in Ruby?

Q4. What is the use of Global Variable in Ruby?

Q6. How is visibility of methods change in Ruby?

Q7. What is a Class Instance Variable

Q8. What are the rules and conventions to be followed in Ruby for naming a method?

Q11. What are the Operators available in Ruby?

Q12. What are the looping structure available in Ruby?

Q13. What is the scope of local variable?

Q14. What are the OOP supported by Ruby?

Q15. If Ruby over PHP, Why?

Q16. Garbage collection in Ruby?Q20. What kind of conditions ruby support?

Q1. What is Agile methodology? What are their Processes?

Q2. Is there any technology apart from agile which we can use?

Q3. What are the servers supported by ruby on rails application?

Q4. What is new in Rails 3.0?

We have a bunch of new features and improved APIs

#### 1.1 Rails 3 requires at least Ruby 1.8.7

ails 3.0 requires Ruby 1.8.7 or higher. Support for all of the previous Ruby versions has been dropped officially and you should upgrade as early as possible.

Rails 3.0 is also compatible with Ruby 1.9.2.

#### Rails Application object

As part of the groundwork for supporting running multiple Rails applications in the same process, Rails 3 introduces the concept of an Application object. An application object holds all the application specific configurations and is very similar in nature to config/environment.rb from the previous versions of Rails.

Each Rails application now must have a corresponding application object. The application object is defined in config/application.rb. If you’re upgrading an existing application to Rails 3, you must add this file and move the appropriate configurations from config/environment.rb toconfig/application.rb.

#### Dependencies and config.gem

The config.gem method is gone and has been replaced by using bundler and a Gemfile, see

#### Vendoring Gems

Rails now uses a Gemfile in the application root to determine the gems you require for your application to start. This Gemfile is processed by the [Bundler](http://github.com/carlhuda/bundler), which then installs all your dependencies. It can even install all the dependencies locally to your application so that it doesn’t depend on the system gems.

, so if you want to use Rails 3 with 1.9.x jump on 1.9.2 for smooth sailing.

Q5. What is Meta-programming? How you are using it inside your rails application?

Everyone in the Ruby world seems to be talking about metaprogramming—how you can use it to remove duplication in your code and write elegant, beautiful programs.

a collection of techniques and tricks known as metaprogramming.

Q7. What is TDD and BDD?

Hint: Test-Driven-Development and Behavior-Driven-Development

Q8. What isrspec, cucumber and Watir? And what it has to do with TDD and BDD?

Though RubyMine provides RSpec test template by default, the complete RSpec support only becomes available, when rspec gem is attached to your Ruby project. RubyMine'sRSpec support, in particular, includes, you’ll learn about testing your Ruby code with [Rspec](http://rspec.info/), one of the best testing libraries in the business.. Along the way, Ruby has spawned a variety of testing tools and frameworks. The latest, AslakHellesoy's Cucumber, is the latest addition to the RSpec family of tools.

http://guides.rubyonrails.org/getting\_started.html

1. Why Ruby on Rails?

Ans: There are lot of advantages of using ruby on rails

1. DRY Principal

2. Convention over Configuration

3. Gems and Plugins

Plugins are just libraries loaded from a specific directory, gems are loaded via Bundler or RubyGemsdirectly.Where this really makes a differences is maintenance and management. What happens when you want the latest and greatest authlogicplugin, well you need to update the files in your directory. That doesn't sound so bad when it's one plugin, but what about something that constantly updates? There was/is an existing system for code packaging and distribution (RubyGems), which lends itself to managing such things.Consider the authlogic example again, what happens if the new version requires some other dependency now? With RubyGems the gem file explicitly defines that relationship, the plugin system does not and such a definition would've been redundant.

4. Scaffolding

**Scaffolding** is a [meta-programming](http://en.wikipedia.org/wiki/Meta-programming) method of building [database](http://en.wikipedia.org/wiki/Database)-backend [software applications](http://en.wikipedia.org/wiki/Software_application). It is a technique supported by some [model-view-controller](http://en.wikipedia.org/wiki/Model-view-controller)[frameworks](http://en.wikipedia.org/wiki/Software_framework), in which the programmer may write a specification that describes how the application database may be used

5. Pure OOP Concept

6. Rest Support

3 REST

Rest stands for Representational State Transfer and is the foundation of the RESTful architecture

HTTP and REST

7. Rack support

Supporting Rake version: 0.8.3

This package contains Rake, a simple ruby build program with capabilities similar to make.

Rake has the following features:

•Rakefiles (rake‘s version of Makefiles) are completely defined in standard Ruby syntax. No XML files to edit. No quirky Makefile syntax to worry about (is that a tab or a space?)

•Users can specify tasks with prerequisites.

•Rake supports rule patterns to synthesize implicit tasks.

•Flexible FileLists that act like arrays but know about manipulating file names and paths.

•A library of prepackaged tasks to make building rakefiles easier.

8. Action Mailer

Action Mailer allows you to send emails from your application using a mailer model and views. So, in Rails, emails are used by creating mailers that inherit from ActionMailer::Base and live in app/mailers. Those mailers have associated views that appear alongside controller views in app/views.

9. Rpc support

Introduction

Ruby RPC is an alternative to DRb. It provides a similar service, but in a way that is slightly more compatible with my way of thinking. It dispenses with the URL syntax and some of the complexity of DRb and adds automatic distributed reference counting garbage collection. This prevents the problem seen in DRb where objects disappear when there are no local references to them. The interface is also designed to be simple and transparent, with a minimum of programmer work needed to use it.

Ruby RPC was developed and is maintained by Brian Ollenberger. If you need to contact me, see the contact link above.

Features

Ruby RPC has the following features and characteristics:

•Transparent interface.

•Allows serving a single object on a single TCP port.

•Allows other arbitrary sockets, such as pipes, UNIX sockets or SSL sockets.

•Provides distributed reference counting garbage collection.

•Fully supports calling remote methods with blocks, totally transparently.

•Automatically optimizes away many unnecessary round trips.

•Works with another project, of mine, Ruby ODB, to provide remote access to an object database

10. Rexml Support

Parse XML using Ruby

The typical way to parse XML in Ruby is to use REXML, which comes as part of the standard library.

•Using REXML

•Using XmlSimple

•Further Reading

Using REXML In order to illustarte how to use REXML to parse XML data returned by Yahoo! APIs, let us try to extract

11. etc..

2. Explain about the programming language ruby?

Ruby is the brain child of a Japanese programmer Matz. He created Ruby. It is a cross platform object oriented language. With legacy code it gives you the power of administration and organization tasks. Being open source, it did go into great lengths of development.

3. Explain about ruby names?

Classes, variables, methods, constants and modules can be referred by ruby names

5. What is Session and Cookies?

Ans: Session: are used to store user information on the server side.

cookies: are used to store information on the browser side or we can say client side

Session : say session[:user] = “arunkumar” it remains when the browser is not closed

6. What is request.xhr?

Ans: A request.xhr tells the controller that the new Ajax request has come, It always return Boolean values (TRUE or FALSE)

7. What is MVC? and how it Works?

Ans: MVC tends for Model-View-Controller, used by many languages like PHP, Perl, Python etc. The flow goes like this: Request first comes to the controller, controller finds and appropriate view and interacts with model, model interacts with your database and send the response to controller then controller based on the response give the output parameter to view, for Example your url is something like this:

http://localhost:3000/users/new

here users is your controller and new is your method, there must be a file in your views/users folder named new.html.erb, so once the submit button is pressed, User model or whatever defined in the rhtmlform\_for syntax, will be called and values will be stored into the database.

13. How you run your Rails Application without creating database ?

Ans: You can run application by uncomment the line in environment.rb

Path =>rootpathconf/ environment.rb

# Skip frameworks you're not going to use (only works if using vendor/rails)

config.frameworks -= [ :action\_web\_service, :action\_mailer,:active\_record ]

14. How to use sqldb or mysql db. without defining it in the database.yml

Ans: You can use ActiveRecord anywhere!

require'rubygems'

require'active\_record'

ActiveRecord::Base.establish\_connection({

:adapter =>'postgresql',

:user =>'foo',

:password =>'bar',

:database =>'whatever'

})

Q18. Suppose in one of my method I am updating the attributes of table, in my model I have defined after\_create do X, and after\_save do Y. Which method will be called?

after\_save

Q2. What is the use of Destructive Method?

* In ruby, we conventionally attach '!' or '?' to the end of certain method names.
* The exclamation point (!, sometimes pronounced aloud as "bang!") indicates something potentially destructive, that is to say, something that can change the value of what it touches.
* ?chop!?affects a string directly, but?chop?with no exclamation point works on a copy.

Here is an illustration of the difference.  
s1 = "forth"  
s1.chop!?  
s2=s1.chop

**How do you write to STDOUT in Ruby?**

Actually two methods are available:

puts writes with a newline

print writes without a newline

**How do you comment out a block of code?**  
=begin  
def my\_commented\_out\_method  
end  
=end  
You could use successive # signs, but that?s just tedious:  
#  
# def my commented\_out\_method  
# end  
#

**Explain about interpolation?**

* Interpolation is a very important process in Ruby.
* Interpolation is the process of inserting a string into a literal.
* There is only one way in which you can interpolate a string into a literal by placing a Hash (#) within {} open and close brackets.
* This refers to a new nameto by referring to the copy of the original method

**How does ruby deal with extremely large numbers?**

* Unlike other programming languages ruby deals with extremely large numbers it doesn?t have any barriers.
* There is no limit on the extent of limit of number usage.
* Ruby performs this function with two different classes they are fixnum and bignum
* Fixnum represents easily managed small numbers and bignum represents big numbers.
* Ruby entirely handles the functioning of these two classes which leaves a programmer to concentrate on his arithmetic operations

**Explain about variables?**

* There are four different types of variables they are local, instance, global, and class.
* Variables can be used in the program without any declaration and they can contain data of any type.
* A local variable contains lower case characters followed by name characters.
* Instance variable starts with a @ sign followed by name characters.

Explain about ruby names?

* Classes, variables, methods, constants and modules can be referred by ruby names.
* When you want to distinguish between various names you can specify that by the first character of the name.
* Some of the names are used as reserve words which should not be used for any other purpose.
* A name can be lowercase letter, upper case letter, number, or an underscore, make sure that you follow the name by name characters

**Explain about the defined operator?**

* Define operator defines whether a passed expression is defined or not.
* If the expression is defined it returns the description string or null if the expression is not defined.
* If a variable is defined it gets initialized.
* If method\_call is defined as true then method also gets defined. This is al
* Explain about environment variables present in ruby? so the same case with super and yield.

Explain about environment variables present in ruby?

Following are some of the environment variables used to control the behavior programming of ruby. While programming ENV object lists some of the current variables. RUBYLIB path searches for libraries. Make sure that you separate each path with colons. RUBYOPT passes command line options to Ruby interpreter. There are many more which can be obtained by searching the huge pool of library.

**Explain about portability?**

Ruby language can be ported to many platforms. Ruby programs can be ported to many platforms without any modification to the source code. This feature made the language very useful and highly used by many programmers worldwide. Some of the platforms used are DOS, UNIX, WINDOWS, etc.

What's going on here?

Ruby runtime uses garbage collection. Your code allocates memory as it runs. From time to time garbage collector kicks in, halts the whole system, and cleans up no-longer-referenced memory using mark-and-sweep algorithm.

GC gets triggered by multiple conditions. The one that matters the most is 8M trigger. Every time you allocate 8M of memory GC runs. Complex Rails request can allocate hundreds or even thousands of megabytes of memory, making GC runs dozens of times. Each GC pass takes 50-150ms. You do the math.

It's possible to patch ruby interpreter, increase the triggers and reduce frequency of garbage collection at the cost of additional memory use. This helps, but does not resolve the problem. If there is more garbage, each GC run takes longer. We'll devote a separate post to this later.

•ruby-prof attributes full cost of garbage collection to the method where GC gets triggered, not the methods that allocate the memory. That's like blaming the last straw for breaking the camel's back.

•ruby-prof does not show that garbage collection took place and how much time it took. All you can tell is that some method took a long time.

Explain about the command line options?

* Ruby`s language is executed from the command line like most of the scripting languages. Programming and behavior language environment can be controlled from the interpreter itself.
* Some of the commands which are used are as follows ?
* d, -h, -e prog, -v, -T, -r lib, etc.

**What two delimiters are used for blocks?**

* Curly braces {?}
* ?do???end?

What is the use of load and require in Ruby?

The basic way to tell a ruby to load a file is to require it, e.g. require some\_file will load some\_file.rb if the file has not already been loaded. loadsome\_file.rb will load the file (and execute any code) regardless of if the file has been loaded already.

In 1.9.2 the path that Ruby looks for these files includes:

•A number of directories in your Ruby install.

•Directories of any gems loaded

•Directories you have added to $LOAD\_PATH, Rails adds a number of app directories to this variable. To see exactly what is added run puts $LOAD\_PATH in both irb and a rails console.

To require a file relative to the file doing the requiring use require\_relativesome\_directory/some\_file.

Require and load both load the file when execution reaches the relevant require or load call, autoloadsome\_file will not load the file until a module in the file is called.

Rails

In addition to adding app directories to the $LOAD\_PATH, Rails loads all \*.rb files in /config/initializers/ when the application starts.

Rails also has an autoload\_path, If Rails comes across an uninitialized constant it will check any paths in autoload\_path for a file matching the name of the constant e.g. when execution hits an unrecognized constant MyConstant it will check for a file called my\_constant.rb in any paths included in autoload\_paths.

The autoload\_path in 3.0.7 is empty, to add the lib directory to it, add config.autoload\_paths += %W(#{config.root}/lib) to config/application.rb.

If the module name does not match the file name then the file will need to be required in the normal way.

Rails also allows you to require a file with require\_dependency. Require\_dependency works exactly like Ruby’s require except, in development mode, it reloads the file when a view is refreshed allowing you to make changes to the required file and see them by refreshing the browser.

Incude and Extend

There seems to be some confusion between require and include/extend but they are totally unrelated. Require and similar deal with loading files from the filesystem. Include and extend deal with including code from one module in another. A class the includes a module allows an instance of the class to access code from the included module. A class that extends allows the class to access code from the module called

Fixtures are a way of organizing data that you want to test against; in short, sample data.

They come in 3 flavors:

1. YAML fixtures

2. CSV fixtures

3. Single-file fixtures

What are the rules and conventions to be followed in Ruby for naming a method?

* By convention method names begin with lower case letter (they can begin with the upper case letter but that makes it look like a constant).
* When a method name has more than one word then the usual convention is to use underscore (e.glike\_this).
* Method names may end with a question mark, equasl sign and sign of excalmation.
* The first convention is the method which ends with question mark returns a value that answers the question posed by the method invocation.
* The second convention is any method which ends with sign of excalmation must be used with caution. Often the mehods which end with excalmation mark are mutators.

Q12. What are the looping structure available in Ruby?

What are the Operators available in Ruby?

Ruby Arithmetic Operators

Ruby Comparison Operators:

Ruby Assignment Operators:

Ruby P Ruby Bitwise Operators:

parallel Assignment:

Ruby Logical Operators:

Ruby Ternary operator:

Ruby Range operators:

If Ruby over PHP, Why?

Environment Variables in Ruby?

Ruby Pseudo-Variables:

They are special variables that have the appearance of local variables but behave like constants.

You can not assign any value to these variables.

self: The receiver object of the current method.

true: Value representing true.

false: Value representing false.

nil: Value representing undefined.

\_\_FILE\_\_: The name of the current source file.

\_\_LINE\_\_: The current line number in the source file.

Q18. What are Float, Dig and Max?

* Float class is used whenever the function changes constantly.
* It acts as a sub class of numeric.
* They represent real characters by making use of the native architecture of the double precision floating point.
* Max is used whenever there is a huge need of Float.
* Dig is used whenever you want to represent a float in decimal digits.

Q20. What kind of conditions ruby support?

if, elsif, else, unless, the ternary operator, case statements

Difference between puts and print

* Puts adds a newline to the end of the output. Print does not.
* I discovered something a bit peculiar about the puts and print methods in Ruby. puts seems to flush immediately, and therefore shows up on $stdout right away. Take the code example below:

5.times {

puts"."

sleep2

}

This functions exactly the way that you would expect. It places a single period on $stdout, followed by a two second pause for five iterations. puts inserts a new line character as well, so instead of placing each period on the same line, each one is on a new line. print does not insert the automatic newline sequence, so it would place each one on the same line. However, the code below does not function the way you would expect.

5.times {

print"."

sleep2

}

The code above waits for 10 seconds and then prints all 5 periods. As it turns out, this is because the print method buffers the output. The easiest way to get around this (for a situation like the above) is to set the sync property on $stdout.

STDOUT.sync = true

5.times {

print"."

sleep2

}

This sets $stdout to avoid buffering the input, which most modern operating systems do. If you find yourself in a situation where you need to have a small amount of output sent immediately to the screen, this is a good technique to utilize to serve this requirement.

Q1. What is Agile methodology? What are their Processes?

Q2. Is there any technology apart from agile which we can use?

Q3. What are the servers supported by ruby on rails application?

Q4. What is new in Rails 3.0?

* We have a bunch of new features and improved APIs
* rails 3.0 requires Ruby 1.8.7 or higher.
* Support for all of the previous Ruby versions has been dropped officially and you should upgrade as early as possible.
* Rails 3.0 is also compatible with Ruby 1.9.2.

Rails Application object

* As part of the groundwork for supporting running multiple Rails applications in the same process,
* Rails 3 introduces the concept of an Application object.
* An application object holds all the application specific configurations and is very similar in nature to config/environment.rb from the previous versions of Rails.
* Each Rails application now must have a corresponding application object.
* The application object is defined in config/application.rb.
* If you’re upgrading an existing application to Rails 3, you must add this file and move the appropriate configurations from config/environment.rb toconfig/application.rb.

Dependencies and config.gem

The config.gem method is gone and has been replaced by using bundler and a Gemfile

#### Vendoring Gems

* Rails now uses a Gemfile in the application root to determine the gems you require for your application to start.
* This Gemfile is processed by the [Bundler](http://github.com/carlhuda/bundler), which then installs all your dependencies.
* It can even install all the dependencies locally to your application so that it doesn’t depend on the system gems., so if you want to use Rails 3 with 1.9.x jump on 1.9.2 for smooth sailing.

What is Meta-programming? How you are using it inside your rails application?

how you can use it to remove duplication in your code and write elegant, beautiful programs.a collection of techniques and tricks known as metaprogramming.

What is has\_many?

* It is a way of defining relationships among models.
* DRY – “Don’t Repeat Yourself” – suggests that writing the same code over and over again is a bad thing.

Convention Over Configuration – means that Rails makes assumptions about what you want to do and how you’re going to d o it, rather than requiring you to specify every little thing through endless configuration files.

REST is the best pattern for web applications – organizing your application around resources and standard HTTP verbs i s the fastest way to go

Q7. What is TDD and BDD?

Test-Driven-Development and Behavior-Driven-Development

What isrspec, cucumber and Watir? And what it has to do with TDD and BDD?

* Though RubyMine provides RSpec test template by default, the complete RSpec support only becomes available, when rspec gem is attached to your Ruby project.
* RubyMine'sRSpec support, in particular, includes
* , you’ll learn about testing your Ruby code with [Rspec](http://rspec.info/), one of the best testing libraries in the business.
* . Along the way, Ruby has spawned a variety of testing tools and frameworks. The latest, AslakHellesoy's Cucumber, is the latest addition to the RSpec family of tools.

Water :Web Application Testing in Ruby

# Automated testing that doesn’t hurt

* Watir, [pronounced ***water***,](http://dictionary.reference.com/browse/water) is an open-source (BSD) family of Ruby libraries for automating web browsers.
* It allows you to write tests that are easy to read and maintain. It is simple and flexible.
* Watir drives browsers the same way people do.
* It clicks links, fills in forms, presses buttons. Watir also checks results, such as whether expected text appears on the page.

Ruby Methods Part 4 - Calling Methods

The dot operator

The send method

The method object

Comparing Code Execution

* The real difference between the Ruby and Java programming languages lie in the method of executing code.
* Java code is first translated to a virtual machine
* Code executed on this virtual machine executes faster than Ruby's interpreted code.
* With the exception of JRuby--which is itself implemented in Java--Ruby's current implementations are interpreted without translating to a virtual machine language first.

Differences in Syntax

* The Java programming language follows a strict c-like syntax whereas Ruby's loose syntax frequently allows for things to be omitted.
* A good way to look at this difference is to compare the grammatical rules you'd have to follow when writing a term paper versus the freedom of language that text messaging provides.

Similarities between Java and Ruby

As with Java, in Ruby…

* memory is managed for you via a garbage collector.
* there’s public, private, and protected methods.
* you’ve got embedded doc tools (Ruby’s is called RDoc).
* The docsgenerated by rdoc look very similar to those generated by javadoc.
* RDoccan produce fairly good content even if the source contains no comments.

Differences between Java and Ruby

* you don’t need to compile your code. You just run it directly.
* there’s different GUI toolkits. Ruby users can try WxRuby, FXRuby, or the bundled-in Ruby Tk for example.
* you use the end keyword after defining things like classes, instead of having to put braces around blocks of code.
* you have require instead of import.
* all member variables are private. From the outside, you access everything via methods.
* parentheses in method calls are usually optional and often omitted.
* everything is an object, including numbers like 2 and 3.14159.
* Classesare objects! For example, Array is a constant name that is bound to theArray class object.
* To create a new object, we call new on the classobject as in a = Array.new
* there are no primitives or data types
* variable names are just labels (not objects). They don’t have a type associated with them.
* there’s no type declarations.
* You just assign to new variable names as-needed

and they just “spring up” (i.e. a = [1,2,3] rather than int[] a = {1,2,3};).

* it’sfoo = Foo.new(“hi”) instead offoo = new Foo( “hi” ).
* the constructor is always named initialize instead of the name of the class.
* you have “mixin’s” instead of interfaces.
* YAML tends to be favoured over XML.
* it’s nil instead of null. Also, nil is a normal object; you can never get a null pointer error!
* there is no method overloading.
* it’s much more common to put many classes in the same file.

Blocks and the Times Loop

1. The first use a Ruby programmer is likely to encounter is the times loop.
2. The times loop is nothing but a method of the Integer class.
3. It's used as a small, primitive loop that will repeat itself a set number of times.
4. The block, which will be called repeatedly, is passed as a single argument.
5. Here, the Ruby program uses the times loop to print a message a set number of times.

#!/usr/bin/env ruby

5.times do

puts"Blocks are powerful"

end

Where's the Block?

* The block is the portion of code between the do and end keywords.
* The curly braces { and } may be used as well, since these are the same as the do and end keywords.
* All the code between these keywords is assembled into an anonymous method (a method without any name).
* The Ruby interpreter remembers this method and, when the times method is called, passes this nameless method to the times method.
* Blocks used in this way emulate a type of loop.

#!/usr/bin/env ruby

array = [ "Bob", "Jim", "Joe"]

array .each do |name|

puts name

end

Passing Arguments

The previous example introduced another feature of blocks--the ability to pass arguments. The names between the pipe characters after the do keyword are the arguments. They're just like the named arguments to any other method, but have a slightly different syntax. The each loop will visit each of the elements in the array in order and call the block for each element, passing the current element as a parameter to that block.

Using Blocks

Writing methods that can accept blocks is simple. There's no special syntax needed, you can simply write the method as you would any other. To call any block passed to the method, use the yield keyword. Like any other method call, when you use the yield keyword you can leave it as it is and call the block without any parameters. Or, if you wish, you can add a parameter list as you would with a method call. The following example illustrates a method that re-implements the times method. The defined method takes a single named argument, the number of times the block should be called. The block argument is invisible, but the yield keyword knows about it and will call the block correctly.

#!/usr/bin/env ruby

defthis\_many\_times(num)

counter = 0

while counter < num

yield

counter += 1

end

end

this\_many\_times(5) do

puts"Blocks are powerful"

end