<Codingcompiler/>

Advanced Puppet Interview Questions And Answers

Top 57 Advanced **Puppet Interview Questions** And Answers For Experienced 2018. Here coding compiler listed frequently asked devops puppet interview questions. These 57 *puppet real time interview questions* are prepared by the industry experts, so this list will help you to crack your next **puppet** devops job interview. All the best for your future and happy learning.

Puppet Interview Questions

1. What is Puppet?
2. What is **chef and puppet** used for?
3. How Puppet works?
4. What are Resources in Puppet?

5. What are Resource types in Puppet?
6. How can you add new resource types to Puppet?
7. What is a Class in Puppet?
8. What is Node definition in Puppet?
9. What are **facts in Puppet**?
10. Puppet can access which facts?
11. What are function in Puppet?
12. What is a class in Puppet, explain with example?
13. How can you configure systems with Puppet?
14. What are Catalogs in Puppet?
15. Explain the agent/master **architecture in Puppet**?
16. How can Puppet agent nodes and Puppet masters communicate with each other?
17. Can you explain stand-alone architecture in Puppet?
18. How can you explain **installing Puppet** agent in Linux?
19. What is Puppet codedir?
20. Where do you find codedir in Puppet?
21. In Puppet where codedir is configured?
22. What is a main manifest or site **manifest in Puppet**?
23. What is Puppet apply?
24. What is **modulepath in Puppet**?
25. What is base modulepath?

Puppet Interview Questions And Answers

Puppet Question # 1) What is Puppet?

Answer # Puppet is an **open-source** software configuration management tool. It runs on many Unix-like systems as well as on Microsoft Windows, and includes its own declarative language to describe system configuration.

Puppet Question # 2) What is **chef and puppet used for?**

Answer # **Puppet and Chef** are the major configuration management systems on Linux, along with CFEngine, Ansible. More than a configuration management

tool, Chef, along with **Puppet** and Ansible, is one of the industry's most notable Infrastructure as Code (IAC) tools.

Puppet Question # 3) How Puppet works?

Answer # It works like this. **Puppet agent** is a daemon that runs on all the client servers(the servers where you require some configuration, or the servers which are going to be managed using puppet.) All the clients which are to be managed will have puppet agent installed on them, and are called *nodes in puppet*.

Puppet Question # 4) What are Resources in Puppet?

Answer # Resources are the fundamental unit for modeling system configurations. Each **resource** describes some aspect of a system, like a specific service or package.

A resource declaration is an expression that describes the desired state for a resource and tells Puppet to add it to the catalog. When Puppet applies that catalog to a target system, it manages every resource it contains, ensuring that the actual state matches the desired state.

Puppet Question # 5) What are Resource types in Puppet?

Answers # Every resource is associated with a resource type, which determines the kind of configuration it manages.

Puppet has many built-in resource types, like files, cron jobs, services, etc. See the resource type reference for information about the built-in resource types.

Puppet Question # 6) How can you add new resource types to Puppet?

Answer # Yes we can also add new resource types to Puppet:
Defined types are lightweight resource types written in the **Puppet language**.

Custom resource types are written in Ruby, and have access to the same capabilities as Puppet's built-in types.

Puppet Question # 7) What is a Class in Puppet?

Answer # Classes are named blocks of **Puppet code** that are stored in modules for later use and are not applied until they are invoked by name. They can be added to a node's catalog by either declaring them in your manifests or assigning them from an ENC.

Classes generally configure large or medium-sized chunks of functionality, such as all of the packages, config files, and services needed to run an application.

Puppet Question # 8) What is Node definition in Puppet?

Answer # A node definition or node statement is a block of Puppet code that will only be included in matching nodes' catalogs. This feature allows you to assign specific configurations to specific nodes.

Puppet Question # 9) What are facts in Puppet?

Answer # Before requesting a catalog (or compiling one with **puppet apply**), Puppet will collect system information with Facter. Puppet receives this information as facts, which are pre-set variables you can use anywhere in your manifests.

Puppet Question # 10) Puppet can access which facts?

Answer # Puppet can access the following facts:
Facter's built-in core facts. Any custom facts or external facts present in your modules.

Top Puppet Interview Questions

Puppet Interview Question # 11) What are functions in Puppet?

Answer # You can write your own functions in the Puppet language to transform data and construct values. A function can optionally take one or more parameters as arguments. A function returns a calculated value from its final expression.

Function Syntax in Puppet:

```
function <MODULE NAME>::<NAME>(<PARAMETER LIST>) >> <RETURN TYPE> {  ...
body of function ...  final expression, which will be the returned value
of the function}
```

Function Example in Puppet:

```
function apache::bool2http(Variant[String, Boolean] $arg) >> String {
case $arg {    false, undef, /(?i:false)/ : { 'Off' }    true, /(?
i:true)/          : { 'On' }    default              : { "$arg" }  }}
```

Puppet Interview Question # 12) What is a class in **Puppet, explain with example?**

Answer # Classes are named blocks of Puppet code that are stored in modules for later use and are not applied until they are invoked by name. They can be added to a node's catalog by either declaring them in your manifests or assigning them from an ENC.

Classes generally configure large or medium-sized chunks of functionality, such as all of the packages, config files, and services needed to run an application.

Defining classes in Puppet:
Defining a class makes it available for later use. It doesn't yet add any resources to the catalog; to do that, you must declare it or assign it from an ENC.

Class Syntax in Puppet:

A class with no parametersclass base::linux { file { '/etc/passwd': owner => 'root', group => 'root', mode => '0644', } file { '/etc/shadow': owner => 'root', group => 'root', mode => '0440', }}

A class with parametersclass apache (String $version = 'latest') { package {'httpd': ensure => $version,

Using the class parameter from above before => File[‘/etc/httpd.conf’], } file {‘/etc/httpd.conf’: ensure => file, owner => ‘httpd’, content => template(‘apache/httpd.conf.erb’),

Template from a module } service {‘httpd’: ensure => running, enable => true, subscribe => File[‘/etc/httpd.conf’], }}

Puppet Interview Question # 13) How can you configure systems with Puppet?

Answer # You can configure systems with Puppet either in a client/server architecture, using the Puppet agent and Puppet master applications, or in a stand-alone architecture, using the Puppet apply application.

Puppet Interview Question # 14) What are Catalogs in Puppet?

Answer # A catalog is a document that describes the desired system state for one specific computer. It lists all of the resources that need to be managed, as well as any dependencies between those resources.
Puppet configures systems in two stages:
Compile a catalog. Apply the catalog.

Puppet Interview Question # 15) Explain the agent/master architecture in Puppet?

Answer # When set up as an agent/master architecture, a **Puppet master server** controls the configuration information, and each managed agent node requests its own configuration catalog from the master.

In this architecture, managed nodes run the **Puppet agent** application, usually as a background service. One or more servers run the Puppet master application, Puppet Server.

Puppet Interview Question # 16) How can Puppet agent nodes and Puppet masters **communicate with each other?**

Answer # Puppet agent nodes and Puppet masters communicate by HTTPS with client verification.

The Puppet master provides an HTTP interface, with various endpoints available. When requesting or submitting anything to the master, the agent makes an HTTPS request to one of those endpoints.

Client-verified HTTPS means each master or agent must have an identifying SSL certificate. They each examine their counterpart's certificate to decide whether to allow an exchange of information.
Puppet includes a built-in certificate authority for managing certificates.

Agents can automatically request certificates through the master's HTTP API. You can use the puppet cert command to inspect requests and sign new certificates. And agents can then download the signed certificates.

Puppet Interview Question # 16) Can you explain stand-alone architecture in Puppet?

Answer # Puppet can run in a stand-alone architecture, where each managed node has its own complete copy of your configuration info and compiles its own catalog.

In this architecture, managed nodes run the Puppet apply application, usually as a scheduled task or cron job. You can also run it on demand for initial configuration of a server or for smaller configuration tasks.

Like the Puppet master application, Puppet apply needs access to several sources of configuration data, which it uses to compile a catalog for the node it is managing.

Puppet Interview Question # 17) How can you explain installing Puppet agent in Linux?

Answer # Install the Puppet agent so that your master can communicate with your Linux nodes.

1. Install a release package to enable Puppet Platform repositories.
2. Confirm that you can run Puppet executables.

The location for Puppet's executables is /opt/puppetlabs/bin/, which is not in your PATH environment variable by default.

The executable path doesn't matter for Puppet services — for instance, service puppet start works regardless of the PATH — but if you're running interactive puppet commands, you must either add their location to your PATH or execute them using their full path.

To quickly add the executable location to your PATH for your current terminal session, use the command export PATH=/opt/puppetlabs/bin:$PATH. You can also add this location wherever you configure your PATH, such as your .profile or .bashrc configuration files.

For more information, see details about file and directory locations.

3. Install the puppet-agent package on your Puppet agent nodes using the command appropriate to your system:
Yum – sudo yum install puppet-agentApt – sudo apt-get install puppet-agentZypper – sudo zypper install puppet-agent

4. (Optional) Configure agent settings.
For example, if your master isn't reachable at the default address, server = puppet, set the server setting to your **Puppet master's hostname**.
For other settings you might want to change, see a list of agent-related settings.

5. Start the puppet service: sudo /opt/puppetlabs/bin/puppet resource service puppet ensure=running enable=true.

6. (Optional) To see a sample of Puppet agent's output and verify any changes you may have made to your configuration settings in step 5, manually launch and watch a Puppet run: sudo /opt/puppetlabs/bin/puppet agent –test

7. Sign certificates on the certificate authority (CA) master.
On the Puppet master:

1. Run sudo /opt/puppetlabs/bin/puppet cert list to see any outstanding requests.

2. Run sudo /opt/puppetlabs/bin/puppet cert sign <NAME> to sign a request.

As each Puppet agent runs for the first time, it submits a certificate signing request (CSR) to the CA Puppet master. You must log into that server to check for and sign certificates. After an agent's certificate is signed, it regularly fetches and applies configuration catalogs from the Puppet master.

Puppet Interview Question # 18) What is Puppet codedir**?**

Answer # Puppet's codedir is the main directory for Puppet code and data. It contains environments (which contain your manifests and modules), a global modules directory for all environments, and your Hiera data.

Puppet Interview Question # 19) Where do you find codedir **in Puppet?**

Answer # Puppet's codedir can be found at one of the following locations: *nix Systems: /etc/puppetlabs/codeWindows: %PROGRAMDATA%\PuppetLabs\code (usually C:\ProgramData\PuppetLabs\code)non-root users: ~/.puppetlabs/etc/code

Puppet Interview Question # 20) In Puppet where codedir is configured?

Answer # The location of the codedir can be configured in puppet.conf with the codedir setting, but note that Puppet Server doesn't use that setting; it has its own jruby-puppet.master-code-dir setting in puppetserver.conf. If you're using a non-default codedir, you must change both settings.

Advanced Puppet Interview Questions

Puppet Interview Questions # 21) What is a main manifest or site manifest in Puppet?

Answer # Puppet always starts compiling with either a single manifest file or a directory of manifests that get treated like a single file. This main starting point is called the main manifest or site manifest.

Puppet Interview Questions # 22) What is Puppet apply?

Answer # The puppet apply command requires a manifest as an argument on the command line. (For example: puppet apply /etc/puppetlabs/code/environments/production/manifests/site.pp.) It can be a single file or a directory of files.

The puppet apply command does not automatically use an environment's manifest. Instead, it always uses the manifest you pass to it.

Puppet Interview Questions # 23) What is modulepath **in Puppet?**

Answer # The Puppet master service and the puppet apply command both load most of their content from modules. (See the page on module structure and behavior for more details.)

Puppet automatically loads modules from one or more directories. The list of directories Puppet will find modules in is called the modulepath.

Puppet Interview Questions # 24) What is base modulepath**?**

Answer # The base modulepath is a list of global module directories for use with all environments. It can be configured with the basemodulepath setting, but its default value is probably suitable for you unless you're doing something unusual.

The default value of the basemodulepath setting is $codedir/modules:/opt/puppetlabs/puppet/modules. (On Windows, it will just use $codedir\modules.)

Puppet Interview Questions # 25) What is SSLdir?

Puppet stores its certificate infrastructure in the ssldir directory. This directory has a similar structure on all Puppet nodes, whether they are agent nodes, Puppet master servers, or the certificate authority (CA) master.

Puppet Labs Interview Questions

Puppet Interview Questions # 26) What the ssldir **directory contains in Puppet?**

Answer # The ssldir directory contains Puppet certificates, private keys, certificate signing requests (CSRs), and other cryptographic documents.
The ssldir directory on Agent nodes and **Puppet masters** contain a private key (private_keys/<certname>.pem), a public key (public_keys/<certname.pem>), a signed certificate (certs/<certname>.pem), a copy of the CA certificate (certs/ca.pem), and a copy of the certificate revocation list (CRL) (crl.pem).

They usually also retain a copy of their CSR after submitting it (certificate_requests/<certname>.pem). If these files don't exist, they are either generated locally or requested from the CA Puppet master.

Puppet Interview Questions # 27) What is cache directory (vardir**) in Puppet?**

Answer # Puppet's cache directory, sometimes called vardir, contains dynamic or growing data that Puppet creates in the course of its normal operations. Some of this data can be mined for interesting analysis, or to integrate other tools with Puppet. Other parts are just infrastructure and can be ignored.

Puppet Interview Questions # 28) Where the vardir **can be stored in Puppet?**

Answer # Location of the vardir directory Puppet Server's cache directory defaults to /opt/puppetlabs/server/data/puppetserver.

The cache directory for Puppet agent and Puppet apply can be found at one of the following locations:
*nix Systems: /var/opt/puppetlabs/puppet/cachenon-root users: ~/.puppetlabs/opt/puppet/cacheWindows: %PROGRAMDATA%\PuppetLabs\puppet\cache (usually C:\Program Data\PuppetLabs\puppet\cache)

Puppet Interview Questions # 29) What are Environments in Puppet?

Answer # Environments are isolated groups of Puppet agent nodes.
A Puppet master serves each environment with its own main manifest and module path. This lets you use different versions of the same modules for

different groups of nodes, which is useful for testing changes to your Puppet code before implementing them on production machines.

Puppet Interview Questions # 30) What are the types of environemts**?**

Answer # The main uses for environments fall into three categories: permanent test environments, temporary test environments, and divided infrastructure.

Puppet Real Time Interview Questions

Puppet Interview Questions # 31) What are Permanent test environments in Puppet?

Answer # In a permanent test environment, there is a stable group of test nodes where all changes must succeed before they can be merged into the production code. The test nodes are a smaller version of the whole production infrastructure.

Puppet Interview Question # 32) What are Temporary test environments in Puppet?

Answer # In a temporary test environment, you can test a single change or group of changes by checking the changes out of version control into the $codedir/environments directory, where it will be detected as a new environment. A temporary test environment can either have a descriptive name or use the commit ID from the version that it is based on.

Puppet Interview Questions # 33) What is Divided infrastructure in Puppet?

Answer # If parts of your infrastructure are managed by different teams that don’t need to coordinate their code, you can split them into environments.

Puppet Interview Questions # 34) What are modules in Puppet?

Answer # Modules are self-contained bundles of code and data. These reusable, shareable units of Puppet code are a basic building block for Puppet.

Nearly all Puppet manifests belong in modules. The sole exception is the main site.pp manifest, which contains site-wide and node-specific code.

Puppet Interview Questions # 35) What is Module layout in Puppet?

Answer # On disk, a module is a directory tree with a specific, predictable structure:
<MODULE NAME> manifests

- files
- templates
- lib
- facts
- examples
- spec
- functions
- types

Puppet Real Time Scenarios

Puppet Interview Question # 36) What are the Types of plug-ins in modules of Puppet?

Answer # Puppet supports several kinds of plug-ins:
Custom facts (written in Ruby).External facts (executable scripts or static data).

Custom resource types and providers (written in Ruby).Custom functions written in Ruby.Custom functions written in the Puppet language.Custom Augeas lenses.Miscellaneous utility Ruby code used by other plug-ins.

Puppet Interview Questions # 37) What is puppet module command?

Answer # The puppet module command provides an interface for managing modules from the Puppet Forge. Its interface is similar to several common package managers (such as gem, apt-get, or yum). You can use the puppet module command to search for, install, and manage modules.

Puppet Interview Questions # 38) Explain the process of installing modules from the command line in Puppet?

Answer # The puppet module install command installs a module and all of its dependencies.

By default, it installs into the first directory in Puppet's modulepath, which defaults to $codedir/environments/production/modules.

For example, to install the puppetlabs-apache module, run:
puppet module install puppetlabs-apache

Puppet Interview Questions # 39) Explain the process of Installing modules from the Puppet Forge?

Answer # To install a module from the Puppet Forge, use the puppet module install command with the full name of the module you want.
The full name of a Forge module is formatted as username-modulename. For example, to instal puppetlabs-apache:
puppet module install puppetlabs-apache

Puppet Interview Questions # 40) How to check the installed modules in Puppet?

Answer # Use the puppet module list command to see which modules you have installed and which directory they're installed in.

To view the modules arranged by dependency instead of location on disk, use the –tree option.

Puppet Master Interview Questions

Puppet Interview Questions # 41) How to uninstalling modules in Puppet?

Answer # Use the puppet module uninstall command to remove an installed module.

Puppet Interview Questions # 42) What are the core commands of Puppet?

Answer # Core commands of Puppet are:

- Pupper Agent
- Pupper Server
- Puppet Apply
- Puppet Cert
- Puppet Module
- Puppet Resource
- Puppet Config
- Puppet Parser
- Puppet Help
- Puppet Man

Puppet Interview Questions # 43) What is Puppet agent?

Answer # Puppet agent manages systems, with the help of a Puppet master. It requests a configuration catalog from a Puppet master server, then ensures that all resources in that catalog are in their desired state.

Puppet Interview Questions # 44) What is Puppet Server?

Answer # Puppet Server compiles configurations for any number of Puppet agents, using Puppet code and various other data sources. It provides the same services as the classic Puppet master application, and more.

Puppet Interview Questions # 45) What is Puppet apply?

Answer # Puppet apply manages systems without needing to contact a Puppet master server. It compiles its own configuration catalog, using Puppet modules and various other data sources, then immediately applies the catalog.

Puppet Interview Questions # 46) What is Puppet cert?

Answer # Puppet cert helps manage Puppet's built-in certificate authority (CA). It runs on the same server as the Puppet master application. You can use it to sign

and revoke agent certificates.

Puppet Interview Questions # 47) What is Puppet module?

Answer # Puppet module is a multi-purpose tool for working with Puppet modules. It can install and upgrade new modules from the Puppet Forge, help generate new modules, and package modules for public release.

Puppet Interview Questions # 48) What is Puppet resource?

Answer # Puppet resource lets you interactively inspect and manipulate resources on a system. It can work with any resource type Puppet knows about.

Puppet Interview Questions # 49) What is Puppet config?

Answer # Puppet config lets you view and change Puppet's settings.

Puppet Interview Questions # 50) What is Puppet parser?

Answer # Puppet parser lets you validate Puppet code to make sure it contains no syntax errors. It can be a useful part of your continuous integration toolchain.

Puppet Command Cheat Sheet

Puppet Interview Questions # 51) What are Puppet help and Puppet man**?**

Answer # Puppet help and Puppet man can display online help for Puppet's other subcommands.

Puppet Interview Questions # 52) What are the sub commands **in Puppet?**

Answer # Puppet's command line tools consist of a single puppet binary with many subcommands. The following subcommands are available in this version of Puppet:

Core Tools: These subcommands form the core of Puppet's tool set, and every user should understand what they do.

1. puppet agent
2. puppet apply
3. puppet cert
4. puppet master
5. puppet module
6. puppet resource
7. puppet lookup

Occasionally Useful Subcommands

Many or most users will need to use these subcommands at some point, but they aren't needed for daily use the way the core tools are.

- puppet config
- puppet describe
- puppet device
- puppet doc
- puppet epp
- puppet help
- puppet man
- puppet node
- puppet parser
- puppet plugin

Niche Subcommands: Most users can ignore these subcommands. They're only useful for certain niche workflows, and most of them are interfaces to Puppet's internal subsystems.

1. puppet ca
2. puppet catalog
3. puppet certificate
4. puppet certificate_request
5. puppet certificate_revocation_list
6. puppet facts

7. puppet filebucket
8. puppet key
9. puppet report
10. puppet status

Unknown or New Subcommands: These subcommands have not yet been added to any of the categories above.

- puppet generate

Puppet Interview Questions # 53) Explain Puppet server?

Answer # Puppet Server is an application that runs on the Java Virtual Machine (JVM) and provides the same services as the classic Puppet master application. It mostly does this by running the existing Puppet master code in several JRuby interpreters, but it replaces some parts of the classic application with new services written in Clojure.

Puppet Interview Questions # 54) What is Hiera?

Answer # Hiera is a key/value lookup used for separating data from Puppet code. Hiera is Puppet's built-in key-value configuration data lookup system.

Puppet's strength is in reusable code. Code that serves many needs must be configurable: put site-specific information in external configuration data files, rather than in the code itself.

Puppet Interview Questions # 55) Why Puppet uses Hiera?

Answer # Puppet uses Hiera to do two things:
Store the configuration data in key-value pairsLook up what data a particular module needs for a given node during catalog compilation.This is done via:

Automatic Parameter Lookup for classes included in the catalog

Explicit lookup calls

Puppet Interview Questions # 56) What is PSON in Puppet?

Answer # PSON is a variant of JSON that puppet uses for serializing data to transmit across the network or store on disk. Whereas JSON requires that the serialized form is valid unicode (usually UTF-8), PSON is 8-bit ASCII, which allows it to represent arbitrary byte sequences in strings.

Puppet uses the MIME types "pson" and "text/pson" to refer to PSON.

Puppet Interview Questions # 57) How PSON is different from JSON?

Answer # PSON does not differ from JSON in its representation of objects, arrays, numbers, booleans, and null values. PSON does serialize strings differently from JSON.A PSON string is a sequence of 8-bit ASCII encoded data. It must start and end with " (ASCII 0x22) characters.

RELATED INTERVIEW QUESTIONS

1. DB2 Interview Questions
2. AnthillPro Interview Questions
3. Angular 2 Interview Questions
4. Hibernate Interview Questions
5. ASP.NET Interview Questions
6. PHP Interview Questions
7. Kubernetes Interview Questions
8. Docker Interview Questions
9. CEH Interview Questions
10. CyberArk Interview Questions
11. Appian Interview Questions
12. Drools Interview Questions
13. Talend Interview Questions
14. Selenium Interview Questions
15. Ab Initio Interview Questions
16. AB Testing Interview Questions
17. Mobile Application Testing Interview Questions
18. Pega Interview Questions

19. UI Developer Interview Questions
20. Tableau Interview Questions
21. SAP ABAP Interview Questions
22. Reactjs Interview Questions
23. UiPath Interview Questions
24. Automation Anywhere Interview Questions
25. RPA Interview Questions
26. RPA Blue Prism Interview Questions
27. Ranorex Interview Questions
28. AWS Interview Questions
29. SSRS Interview Questions
30. SQL Interview Questions

Coding Compiler / December 16, 2017 / Interview Questions, Puppet Interview Questions / Devops, Interview Questions, Puppet, Puppet Interview Questions

1 thought on “Advanced Puppet Interview Questions And Answers”

BestDeb

February 16, 2018 at 11:38 am

I have noticed you don’t monetize your site, don’t waste your traffic, you can earn additional bucks every month because you’ve got hi
quality content. If you want to know how to make extra bucks,
search for: Ercannou’s essential tools best adsense alternative