

For Platform Administrators

This part of our documentation serves as a reference guide for IT staff and those installing and operating the Apprenda Platform. Whether you're running Apprenda on a single workstation or 100+ servers in your enterprise data center, this section contains information on installing, configuration, troubleshooting, and managing Apprenda:

Extending the Apprenda Platform

Guest Applications as Service-oriented Extensions

The Apprenda platform runtime exposes a dynamically configurable service oriented extension system that allows developers to publish guest applications with functionality that extends the default platform business workflows and then tell the platform to call those services during normal workflow operations. The platform API contains service contracts for four core workflow extension points, each responsible for separate business service workflows. By implementing these service contracts in a WCF service or group of services, publishing, and then configuring the platform in real-time, platform owners can achieve a huge amount of customization. For example, you may want to capture information about new users that are created in the system and communicate that information to a 3rd party service. The configuration of these extensions is performed in the Operations Center and is based on Customer Settings, so it is therefore highly dynamic. Because extension services are guest apps on the platform, they gain the benefit of Apprenda's application lifecycle management, allowing platform operators to test variations of their extensions. What's more, multiple extension services can be called in succession, allowing platform operators to craft intended workflows from extension services published by developers.

Developing Extension Services

The service contracts for Apprenda's extension services can be found in the `Apprenda.SaaSGrid.Extensions` namespace, which is located in the `SaaSGrid.API` assembly delivered with the platform SDK. There are four service contracts:

Authentication & Signup Workflow:

`Apprenda.SaaSGrid.Extensions.ICitadelExtensionService`

User Administration Workflow:

`Apprenda.SaaSGrid.Extensions.IAccountPortalExtensionService`

Developer Workflow:

`Apprenda.SaaSGrid.Extensions.IDeveloperPortalExtensionService`

Storefront Workflow:

`Apprenda.SaaSGrid.Extensions.IStoreFrontExtensionService`

Presentation Deployment Workflow:

`Apprenda.SaaSGrid.Extensions.IPresentationExtensionService`

To build a service whose methods will be called by the platform extension points, developers should create WCF services that implement one or more of these contracts. Data sent to extension services uses a set of objects located in the namespace `Apprenda.SaaSGrid.Extensions.DTO`, which is also contained in the `SaaSGrid.API` assembly.

This is an example of a WCF service class that implements the `ICitadelExtensionService`, and will therefore be capable of handling Authentication & Signup hooks:

```
namespace MyApprendaExtensionServices.Citadel
{
    public class ExtensionService : ICitadelExtensionService
    {
        public void OnTenantOnBoarding(TenantOnBoarderDTO tenant)
        {
            // your extension logic here
        }

        public void OnTenantOnBoarded(TenantOnBoarderDTO tenant)
        {
            // your extension logic here
        }
    }
};
```

Along with its accompanying System.ServiceModel configuration in the App.config:

```
<service name="MyApprendaExtensionServices.Citadel.ExtensionService"
    behaviorConfiguration="ApprendaServiceBehavior">
    <endpoint address="net.tcp://localhost:40000/ICitadelExtensionService"
        binding="netTcpBinding"
        bindingConfiguration="DefaultTcpBinding"
        behaviorConfiguration="ApprendaEndpointBehavior"
        contract="Apprenda.SaaSGrid.Extensions.ICitadelExtensionService"/>
</service>
```

This WCF service should be packaged into an archive for deployment like any other platform guest application and promoted through the publishing stages in the Developer Portal.

Configuring the Platform to Use Extension Services

Custom Settings in the Operations Center are used to tell the platform about the extension services to employ during standard workflows. There is a Custom Setting for each of the four core services that can be extended:

Authentication & Signup Workflow Setting Name:

CitadelExtServices

User Administration Workflow Setting Name:

AccountPortalExtServices

Developer Workflow Setting Name:

DeveloperPortalExtServices

Storefront Workflow Setting Name:

StoreFrontExtServices

Presentation Workflow Setting Name:

PresentationExtServices

The values for these settings should be a comma-delimited list of application aliases corresponding with the unique

application and version aliases (for application in the Sandbox stage) , followed by the service class name that is being called.

For example, to configure the platform to call the ExtensionService in a published application with alias 'myextservice' during **account signup**, the ExtensionService must implement `Apprenda.SaaSGrid.Extensions.ICitadelExtensionService`, be configured to expose that contract endpoint, and it must be published. Then, the setting 'CitadelExtServices' should have a value of 'myextservice', or 'myextservice(v1)', which specifies a target version of the application. If developers publish version two with the alias v2, the setting can be updated to 'myextservice(v2)' and that extension service will be called during the next **account signup** workflow that fires. To call both versions of this application, or call multiple application/version pairs during the same workflow, use a comma-delimited, like this:

Setting Name: CitadelExtServices

Setting Value: myextservice(v1)/ExtensionService, myextservice(v2)/ExtensionService, myotherextservice(v1)/ExtensionService

During **account creation**, the contract endpoints for `ICitadelExtensionService` on the WCF service called 'ExtensionService' will be called in order.

Note: Applications in the Published stage do not require a specified version alias, only ones in the Sandbox stage do; if no version is specified at all, it will call whatever version is Published.

Available Hook Points

The following is an explanation of each hook point that an extension service can implement:

Authentication & Signup Workflow Hooks

The following service methods will be called by Apprenda's Citadel service, which handles account onboarding and authentication.

- OnTenantOnBoarding
- OnTenantOnBoarded
- OnUserLogin
- OnUserLogout

User Administration Workflow Hooks

The following service methods will be called by Apprenda's AccountPortal service, which handles customer and end user account management:

- OnCreatingUser
- OnUserCreated
- OnRemovingUser
- OnUserRemoved
- OnUpdatingUser
- OnUserUpdated

Developer Workflow Hooks

The following service methods will be called by Apprenda's DeveloperPortal service, which handles guest application publishing:

- OnPromotingVersion
- OnVersionPromoted
- OnDemotingVersion

- OnVersionDemoted

Storefront Workflow Hooks

The following service methods will be called by Apprenda's Storefront service, which handles the platform's authorization mechanisms such as subscriptions:

- OnPurchasingSubscription
- OnSubscriptionPurchased
- OnActivatingSubscriptions
- OnSubscriptionsActivated
- OnExpiringSubscriptions
- OnSubscriptionsExpired
- OnSuspendingSubscriptions
- OnSubscriptionsSuspended
- OnCancellingSubscriptions
- OnSubscriptionsCancelled

Presentation Workflow Hooks

The following service methods will be called by Apprenda's DeveloperPortal service during deployment of UI components:

- FilterHostsForPartition
- FilterHostsForTenantShard

Retrieving Contextual Information

The Apprenda Platform allows developer to access contextual information regarding the current organization (refer to the current link for more information: <http://docs.apprenda.com/current/contexts>). When using extensions a new TenantContext is appended to the stack for the organization that developed the extension, this means that TenantContext.Current won't retrieve the running organization but the developer. In order to retrieve the right information you can use the following method:

```
private TenantContext EstablishProperContext()
{
    if (TenantContext.Current.History.Skip(1).Any())
    {
        /*Because this will run as a single tenant app, the top tenant context is the
        e tenant ID for that deployed the extension application
        In order to get the ID for the executing tenant, we have to move up the stack
        once.*/
        return TenantContext.NewTenantContext(TenantContext.Current.History.Skip(1).
        First().TenantId);
    }
    else
    {
        return TenantContext.NewTenantContext(TenantContext.Current.TenantId);
    }
}
```

Extensions Example

The example is a simple Apprenda application that implements all four extension interfaces. The application logs an info message everytime each extension method is called.

[Extension Example](#)

Extensions Library and Support

Extensions are an excellent way to enhance the platform functionality to fit your particular need. We dedicated an entire forum just for them in the Apprenda support website. In this forum you will be able to download extensions developed by the Apprenda support team ready to be deployed, ask questions and get general implementation information.

We are always adding new extensions, if you have an idea for one email us at support@apprenda.com. Also, if you have developed an extension yourself that wish to publish for the community, please feel free to post it!

[Extensions Library and Support](#)

The System Operation Center (SOC)


The System Operation Center or SOC is the infrastructure management portal for your Apprenda Environment. The soc is accessible via the following URL:

[http\(s\)://apps.\[your root url here\]/soc/](http(s)://apps.[your root url here]/soc/)

SOC Dashboard

The SOC Dashboard provides platform owners with an overview of the current state of the platform. By default the only information available are the last 5 **Event Logs** recorded by the system.

Note: The SOC Dashboard **Recent Events** section will only show Error or Fatal logs of Guest applications or event logs generated by Apprenda services.

 Infrastructure Applications Repository Browser Event Logs Platform Access System Configuration 				
Recent Events View Logs Hide				
Level	Timestamp	Source	Machine	Text
DEBUG	11/15/2011 01:50:58 PM	Apprenda.SaaSGrid.API.Live.AuthenticationInitMo	ERIK-VM2	Requested Url: https://apps.apprenda.cstest2/developer/Images/P... Host Header: apps.apprenda.cstest2
DEBUG	11/15/2011 01:50:56 PM	Apprenda.SaaSGrid.ProviderPortal.Service.IProvid	ERIK-VM2	GetAllInvoicesByStatus(InvoiceStatus invoiceStatus, Int32 pageSize, Int32 pageNumber) returned 'Number of record returning: 0' Searched Row Count: 0'
DEBUG	11/15/2011 01:50:56 PM	Apprenda.SaaSGrid.ProviderPortal.Service.IProvid	ERIK-VM2	GetAllInvoicesByStatus(InvoiceStatus invoiceStatus, Int32 pageSize, Int32 pageNumber) invoiceStatus = 'Unpaid' pageSize = 1' pageNumber = 1'
DEBUG	11/15/2011 01:50:56 PM	Apprenda.SaaSGrid.TenantPortal.ITenantPortal	ERIK-VM2	GetSubscriptionMapping(Guid versionId, Guid tenantId, Guid userId) returned 'ApplicationAlias: 'developer' TenantAlias: 'cstest' ApplicationId: '316b32db-e812-40b8-9be0-9a4292bc95cc' VersionId: '316b32db-e812-40b8-9be0-9a4292bc95cc' TenantId: 'f487ac9b-1bc5-4385-b7e3-ab9aff77edfd' UserId: '6ea8e5fd-6db0-4a84-9bc1-9f9400854fc4' SubscriptionId: '8d7ad018-
DEBUG	11/15/2011 01:50:55 PM	Apprenda.SaaSGrid.TenantPortal.ITenantPortal	ERIK-VM2	GetSubscriptionMapping(Guid versionId, Guid tenantId, Guid userId) versionId = '316b32db-e812-40b8-9be0-9a4292bc95cc' tenantId = 'f487ac9b-1bc5-4385-b7e3-ab9aff77edfd' userId = '6ea8e5fd-6db0-4a84-9bc1-9f9400854fc4'

When **Resource Throttling** is **Enabled**, the SOC Dashboard provides information on the current overall allocation of resources and the top guest applications based on memory and CPU allocation.



Configuring your System

Enabling SSL for the Apprenda Environment

Although you have the option of configuring SSL settings during installation, once Apprenda is already running you can modify SSL enforcement from the **Security** page (accessible from the **System Configuration** menu):

SSL Enforcement

SSL is not enforced for this Apprenda environment. Frontend servers will accept both insecure *HTTP* and secure *HTTPS* connections from clients.

From this page **Enabling/Disabling SSL Enforcement** can be done by clicking the appropriate button.

When SSL enforcement is enabled, all future traffic on HTTP is automatically re-directed to HTTPS, so there is no end-user disruption – even if the user is already logged in to the system. While enabling SSL enforcement requires all user traffic to be re-directed to HTTPS, disabling SSL enforcement does not disable HTTPS or require all user traffic to be re-directed to HTTP. This means that the end-user may always use SSL to access their Apprenda applications even if it is not required.

Updating the SSL Certificate

Although you have the option of configuring the certificate settings during installation, once SaaSGrid is already running you can modify SSL enforcement or update your certificate from the **Security** page (accessible from the **System Configuration** menu):

Frontend SSL Certificate

Certificate (.pfx)

Password

If you have a certificate you would like to upload, locate it with the **Browse** button, type the password and then click **Upload Certificate**. You can also **repair your HTTPS Bindings** by clicking on the button.

Retrieving License Information

The **Licensing** page (accessible through the **System Configuration** menu) displays information about your current Apprenda licensing allowances. If Apprenda is licensed, the license expiration (if there is one) is displayed. In addition to license status, each license for Apprenda governs a set of attributes. Apprenda licenses grant or deny access to these attributes of the system.

Your Apprenda License

Number of Grid Servers

The number of servers allowed to actively participate in this system at one time.

Number of Applications

Controls the total number of applications that can be created per provider account.

Entitlement / Pricebook definitions per Application

The number of entitlement or pricebook definitions allowed per application.

Number of Organizations

The number of organizations allowed.

Number of Users per Organization

The number of users allowed per organization.

Total Memory (GB) of Grid Servers

The total amount of memory, in gigabytes, allowed across all actively participating servers in this system at one time.

Extensions Support

Extensions are the Apprenda platform's extensibility points to customize workflows and tailor behavior.

Applying a New License File

If you have a license file from Apprenda that you need to apply to your environment, you may use the section with the **Do you have a new license file?** header. **Browse** for your license file, and then click the **Upload License** button. Apprenda will apply the new license parameters immediately.

Do you have a new license file?

Browse...

Upload License

Managing the Apprenda Infrastructure

When a Windows server is commissioned to host Apprenda software components, that server will appear in the Servers list on your Infrastructure page. Servers will appear in this list once they are running the Apprenda Physical Host Windows service and have communicated to the rest of the fabric. At that point, the server's resources--such as RAM, CPU, and disk space--are noted as participants in the fabric. This alone does not mean the server will be used by Apprenda's deployment mechanisms. Nonetheless, with the Apprenda Physical Host Windows service running,

Apprenda knows of the server and its resource capabilities.

The list of servers participating in your environment is located on the center of the page. Each item in the list displays the server's name and its roles:

- Frontend – these servers have the requirements to host deployed application user interfaces in IIS 7 or later. These servers must also have an instance of the Apprenda UserInterfaceManager service running in order to be included in Apprenda's UI deployment strategy.
- Services – these servers have the requirements to host deployed application web service instances.
- Databases – these servers have the requirements to host deployed application databases in a supported version of SQL Server. These servers must also have an instance of the Apprenda StorageManager service for local SQL Server instances.

Note: If Resource Throttling is enabled, the **Infrastructure** page will display the memory and CPU capacity and allocation for each server.

On the left side of the Infrastructure page, a display shows where the Core Components of the environment are located:

Core Components

F-APP1 is hosting the following components:

- Licensing Service
- Logger Service
- SMART Service

F-DB1 is hosting the following components:

- Core Database
- Repository

F-WEB1 is hosting the following components:

- Default Web Host
- Load Manager

Connecting Remotely to a Server

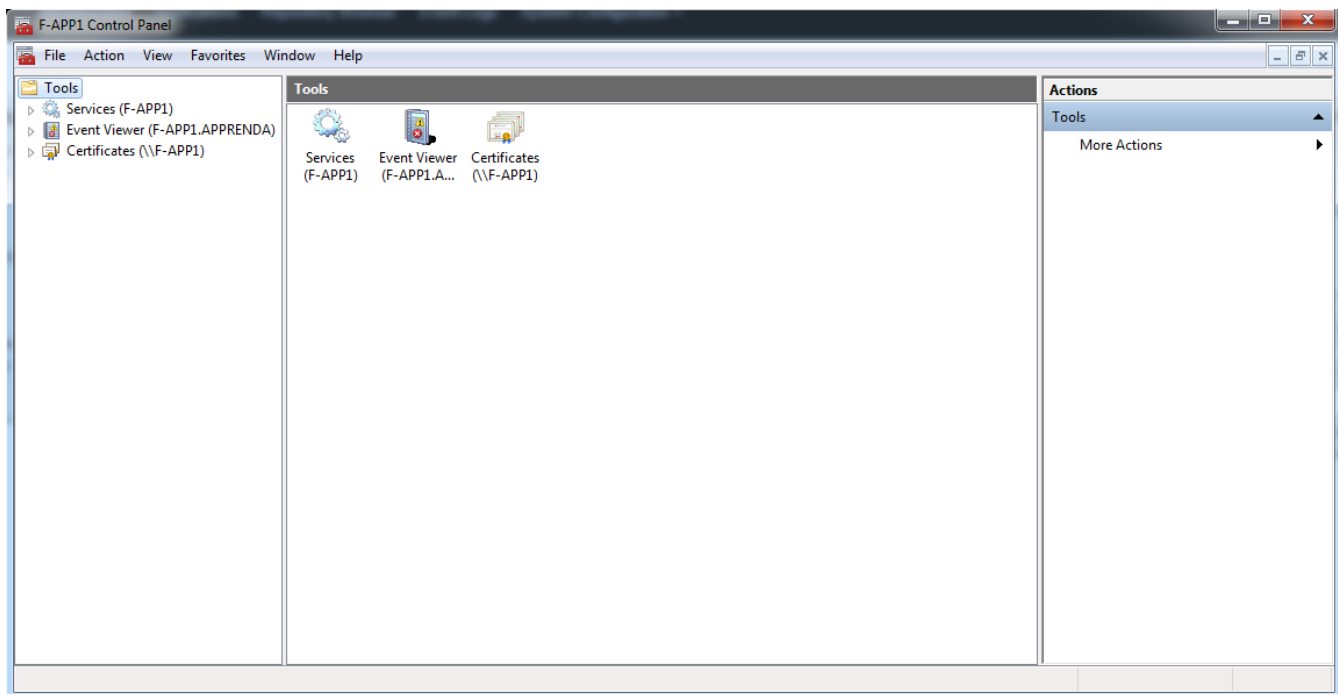
The Apprenda platform allows you to download a Remote Desktop Connection file (rdp) to connect directly to each one of your servers participating in the environment. To download the link, click on the arrow next to the **View Details** button for the server and select **Remote Desktop**.

Name	Roles	Memory Capacity	CPU Capacity	Cores	Architecture	
F-APP1		0MB / 2047MB allocated	0Mhz / 9040Mhz allocated	4	64bit	View Details
F-APP2		0MB / 2047MB allocated	0Mhz / 4520Mhz allocated	2	64bit	View Details
F-DB1		0MB / 2047MB allocated	0Mhz / 4520Mhz allocated	2	64bit	View Details
F-WEB1		2000MB / 2047MB allocated	2000Mhz / 9040Mhz allocated	4	64bit	View Details
F-WEB2		1000MB / 2047MB allocated	1000Mhz / 4520Mhz allocated	2	64bit	View Details

Viewing 1 - 5 of 5

Managing a Server Remotely

The Apprenda platform allows you to download a Management console file (msc) to remotely manage each of your servers participating in the environment. The console loads the following snap-ins: Services, Event Viewer and Certificates. To download the link, click on the arrow next to the **View Details** button for the server and select **Manager Server**.



Managing a Server's Workloads

In Apprenda, a Workload is defined as an instantiation of an Application component currently deployed. There are three types of workloads that can be deployed in the Apprenda Platform:

- Frontend

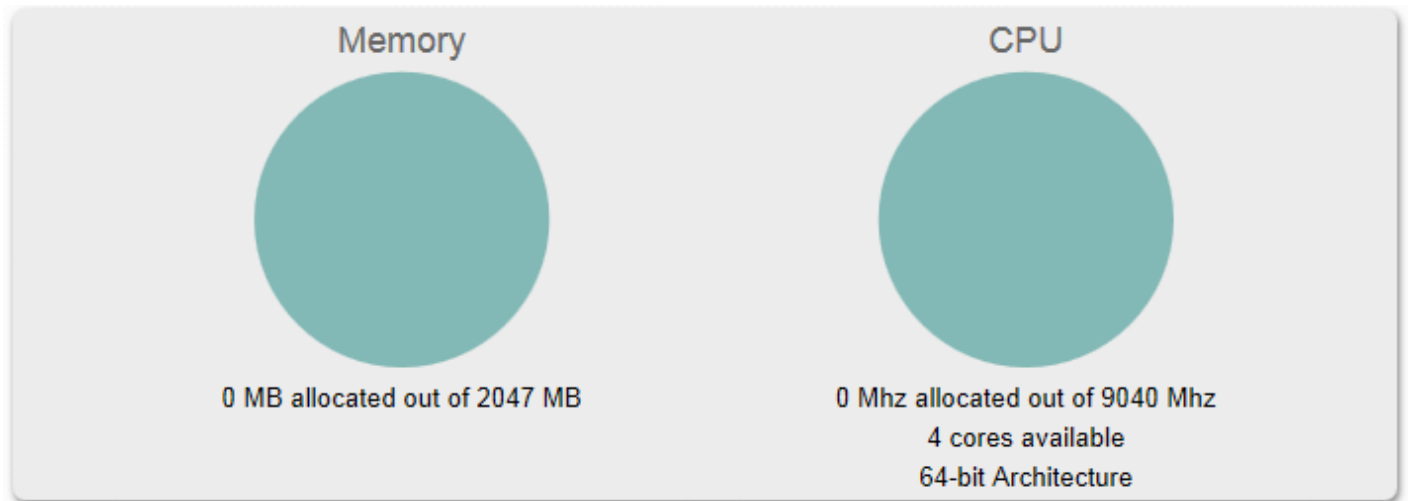
- Services
- Databases

In order to see which Workloads are currently deployed on a server, click on the server name in the **Infrastructure** page. The **Workloads** tab contains a list of all the workloads currently deployed in the server. You can search for a particular item using the **Search** textbox and filter down by:

- Applications
- Components
- Developers
- Policies

Status	Component	Application	Developer	Mem	CPU	Cores	Process Id	Policy	
●	Apprenda SaaSGrid.Chronos.Chror Registration about an hour ago	Scheduling Services Version 1	Apprenda, Inc apprenda			1	3880	Apprenda Core Services	Stop
●	Apprenda SaaSGrid.Citadel.Service Registration about 22 hours ago	Authentication Services Version 1	Apprenda, Inc apprenda			1	2996	Apprenda Core Services	Stop
●	Apprenda SaaSGrid.ProviderPortal Registration 2 days ago	Developer Portal Version 1	Apprenda, Inc apprenda			1	2752	Apprenda Core Services	Stop
●	Apprenda SaaSGrid.SMART.Routin Registration 2 days ago	Routing Services Version 1	Apprenda, Inc apprenda			1	2908	Apprenda Core Services	Stop
●	Apprenda SaaSGrid.TenantPortal.T Registration about an hour ago	Account Portal Version 1	Apprenda, Inc apprenda			1	3664	Apprenda Core Services	Stop

Note: If **Resource Throttling** is **Enabled**, the **Server** information page will show additional graphs displaying the total memory and CPU available and allocated for that particular server. In addition, the **Workloads** tab will display information on the **Resource Policy** assigned to each **Workload** (**Frontend** and **Service** only).



The **Service Workload Registrations** tab shows a list of all the WCF services that registered or unregistered with the local Apprenda **Router** along with a timestamp of when the event happened.

Status	Type	Time ↕	Service Instance
✓	Registration	11/16/2011 12:22:49 PM	Apprenda.SaaSGrid.Chronos.Chronos Instance Id: <i>c2a6a9f0-096d-49ab-be53-58a9eaa14f3c</i>
✓	Registration	11/16/2011 12:21:22 PM	Apprenda.SaaSGrid.TenantPortal.TenantPortalService Instance Id: <i>7c6f0087-d540-4188-96c2-829cd8fea0f5</i>
✓	Registration	11/15/2011 03:34:10 PM	Apprenda.SaaSGrid.Citadel.Service.CitadelService Instance Id: <i>f35f93a6-8acd-42e9-8187-7900602e8dd5</i>
✓	Registration	11/14/2011 01:20:06 AM	Apprenda.SaaSGrid.SMART.Routing.Router Instance Id: <i>dd6b7246-999c-4059-bffd-b6efb1ddb929</i>

Starting and Stopping Service Workloads

At times, it may be necessary to start or stop a single web service instance. To do so, click on the **Stop** button on the right of the **Workload** item or click on the arrow and select **Stop**. To start, click on the **Start** button. Note that taking these actions on a single web service instance only affects that instance. Other instances of the same web service may continue to operate elsewhere on the platform if they are deployed.

Removing a Service Workload

You can remove a **Service** workload by clicking on the arrow on the right end of the Workload item and choosing the **Remove** option. If you choose to remove a **Service workload**, you are not only killing the running process if the web service is started, but you are also instructing Apprenda to remove that instance’s local launchpad (binaries) from the server and unregister any record of the web service instance with the rest of the platform. Doing so will generate an Unregistration event in the **Service Workload Registrations** in the server details.

Viewing Launchpads

You can view the launchpad for a **Service** workload by clicking on the arrow on the right end of the Workload item and choosing the **View Launchpad** option. This will open a new tab and direct you to the **Repository Browser** page, where the selected launchpad information will appear.

Viewing Service Workload Details

You can view the details for a **Service** workload by clicking on the arrow on the right end of the Workload item and choosing the **View Details** option. This will open a pop up window with the following information:

- **Service Type Name & ID** – the type of web service that this web service is running, including that service type’s unique ID. This information will match the information found in Apprenda’s Application Catalog.
- **Topology Address** – The URI that Apprenda uses to route requests to this web service instance
- **Binary** – the actual binary that contains the service contract for the web service type that this web service instance is running. Most likely, this binary was uploaded by you as part of your application archive deployment, or it is a binary for one of Apprenda’s core services.
- **Service Metadata** – information about the provider that created this web service, the application that it belongs to, and the specific application version it belongs to. Note that the same web service type can be deployed for multiple versions of the same application at the same time across the platform. This information is helpful in determining which version of the web service instance is running.

- **Endpoints** – The endpoints that Apprenda has exposed for the web service instance, allowing it to be reached via multiple protocols and channel types. If the web service implements multiple web service contracts, there will be endpoints for each. Each endpoint has a unique URI within the Apprenda topology.

Apprenda.SaaSGrid.SMART.Catalog.CatalogingService
✕

Apprenda.SaaSGrid.SMART.Catalog.CatalogingService

62837D69-DEED-41F4-A596-B4B7009C21AC
Running as PID 3192 on F-APP2

Instance Id: 2be1f324-05f1-4f83-affd-d39d5c62f437
Topology Address: net.tcp://f-app2:33000/2be1f324-05f1-4f83-affd-d39d5c62f437/Topology
Service Binary: SaaSGrid.SMART.Catalog.dll

Created by: Apprenda, Inc
00000000-0000-4000-0000-000000000001
Application: Cataloging Services
161B3EA3-1D2F-45A3-982C-88C1D5E1D2C6
Version: Cataloging Services V1
B58D2B68-E00F-4B28-BE3A-BFE38F6C1516

Note: The platform will not automatically reclaim this instance.

Endpoints:

- 🔗 **Apprenda.SaaSGrid.SMART.Catalog.ICatalogingService**
net.tcp://f-app2:33000/a58cba08-53da-443a-8542-e1b929d4e953
Tcp transport
Binary encoding
- 🔗 **Apprenda.SaaSGrid.SMART.Catalog.IResourceAllocationService**
net.tcp://f-app2:33000/70bb7f74-2ce0-45f3-b4fd-6082c5c96ca4
Tcp transport
Binary encoding
- 🔗 **Apprenda.SaaSGrid.SMART.Catalog.IApplicationInformationService**
net.tcp://f-app2:33000/bb3c3b47-3816-496d-ab02-53e7ba68d35f
Tcp transport
Binary encoding

Managing the Router

Click on the **Router Control Panel** tab in the server details to view that server’s routing profile. Each server runs a message routing process called the Apprenda Router that ushers web service requests around the platform to known endpoints. This is how Apprenda maintains location transparency when deploying application artifacts to various servers on the platform.

To understand the Router Control Panel, it is important to understand what the Apprenda Router does – so we’ll use a small sidebar to illustrate.

Imagine the communication pathway of a standard request to an application GUI: a request arrives at the GUI via an HTTP request and the GUI needs to display data in its response, so it might make a call to a deployed web service in order to obtain information or perform business logic. Instead of the GUI having to be aware of the endpoint address(es) of web service instances for the target service type, the GUI is configured automatically by Apprenda to talk to a local Apprenda Router on the same server. That local router maintains awareness of endpoint URIs across the platform and forwards the message to an appropriate endpoint, waits for a response, and then ushers the response back to the GUI.

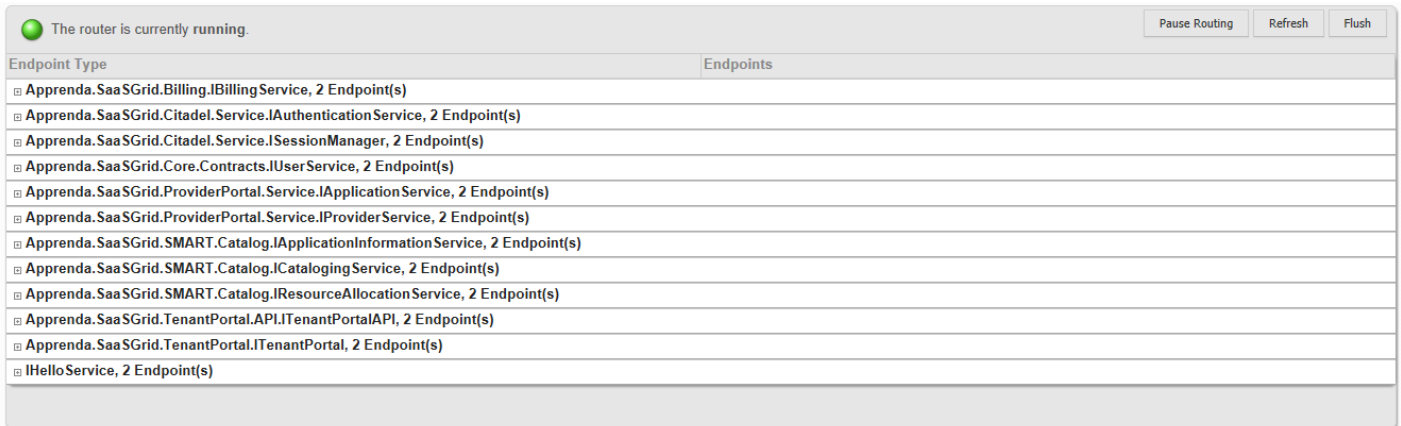
The Router Control Panel, as expected, displays a list of the specific endpoint URIs for which the router has address records. This is akin to viewing routing tables in DNS. This routing table is maintained by the router and Apprenda to ensure that the router has knowledge of any necessary endpoints to which it might need to forward messages.

Clearing Router Targets

Click on the **Flush** button to empty the router's routing table. The router will then begin to auto discover endpoints as requests are made and endpoint URIs are needed to satisfy those requests. This is a good approach to take if web service requests are taking a long time or timing out completely, as that may indicate that the router has stale records in its routing table for endpoints that no longer exist (Note: the router and Apprenda continuously work together to maintain these records in real time, so this shouldn't happen).

Pausing the Router

You can temporarily pause the server's router by clicking the **Pause routing** button. This will not clear the router's targets, but will instruct the router not to forward messages until you click **Start routing**. This technique is helpful in certain debugging scenarios.



The screenshot shows a control panel for the router. At the top left, there is a green status indicator and the text "The router is currently running." To the right of this are three buttons: "Pause Routing", "Refresh", and "Flush". Below this is a table with two columns: "Endpoint Type" and "Endpoints". The table contains the following rows:

Endpoint Type	Endpoints
Apprenda.SaaSGrid.Billing.IBillingService, 2 Endpoint(s)	
Apprenda.SaaSGrid.Citadel.Service.IAuthenticationService, 2 Endpoint(s)	
Apprenda.SaaSGrid.Citadel.Service.ISessionManager, 2 Endpoint(s)	
Apprenda.SaaSGrid.Core.Contracts.IUserService, 2 Endpoint(s)	
Apprenda.SaaSGrid.ProviderPortal.Service.IApplicationService, 2 Endpoint(s)	
Apprenda.SaaSGrid.ProviderPortal.Service.IProviderService, 2 Endpoint(s)	
Apprenda.SaaSGrid.SMART.Catalog.IApplicationInformationService, 2 Endpoint(s)	
Apprenda.SaaSGrid.SMART.Catalog.ICatalogingService, 2 Endpoint(s)	
Apprenda.SaaSGrid.SMART.Catalog.IResourceAllocationService, 2 Endpoint(s)	
Apprenda.SaaSGrid.TenantPortal.API.ITenantPortalAPI, 2 Endpoint(s)	
Apprenda.SaaSGrid.TenantPortal.ITenantPortal, 2 Endpoint(s)	
IHelloService, 2 Endpoint(s)	

Managing your Apprenda Applications

The **Applications** page contains a list of the application currently deployed in the Apprenda Platform in the **Sandbox** or **Published** stage. The main page shows the following information for each application:

- Application name and alias
- Version Deployed
- Stage
- Development Team that published the application
- Workload Type(s) that the application has (**Frontend, Services, Databases**)

Guest Applications

Application	Version	Stage	Development Team	Workload Type(s)
Account Portal alias: account	Version 1 alias: v1	Published	Apprenda, Inc	
Authentication Services alias: authentication	Version 1 alias: v1	Published	Apprenda, Inc	
Billing Services alias: billing	Version 1 alias: v1	Published	Apprenda, Inc	
Cataloging Services alias: catalog	Version 1 alias: v1	Published	Apprenda, Inc	
Common UI Resources alias: resources	Version 1 alias: v1	Published	Apprenda, Inc	
Developer Portal alias: developer	Version 1 alias: v1	Published	Apprenda, Inc	

Managing a Deployed Version

Clicking on the **Version** link of a deployed application takes you into the Application Details page for that particular version. The left side of the details page shows some basic information about the application:

- Version information
- Application Name
- Developer
- Platform Service used by this application (Authentication, Authorization, Multi-tenancy and/or Billing)

Version
Version 1
v1
Published

Application
Account Portal
account

Developer
Apprenda, Inc

Platform Services

- Authentication
- Authorization
- Multi-tenancy

The **Workloads** tab shows the currently deployed **Workloads** for this particular application, the location and in the case of a Service, the Process ID. If **Resource Throttling** is enabled on this environment, each Workflow (except databases) will display the policy attached to them, average CPU usage and average memory usage.

Status	Component	Server	PID	Policy	Avg. CPU Usag	Avg. Memory U	
●	User Interface	F-WEB1		Apprenda Core Services	0% of 9040 MHz	0% of 2047 MB	Undeploy ▼
●	Apprenda.SaaSGrid.TenantPort	F-APP1	3664	Apprenda Core Services	0% of 9040 MHz	5.12% of 2047 MB	Stop ▼
●	Database	F-DB1					
●	Provider Database	F-DB1					

Managing Frontend Workloads

The Version Details page allows for the management of individual **Workloads**. For **Frontend Workloads**, there are two actions that can be taken:

- Undeploy - The Undeploy action allows the platform owner to uninstall an instance of this version's website.
- Move - The Move action allows the platform owner to move the website to a different server. When moving a **Frontend Workload** you can let Apprenda choose a suitable server for you (based on role and resource allocation) or choose it yourself.

Move a Frontend Workload

Workload: User Interface

The workload will be moved to a new server. *This will result in a new process on the target server.*

Server: Let Apprenda Choose a Server

The platform will select a suitable server to run the new workload.

Managing a Service Workload

The Version Details page allows for the management of individual **Workloads**. For **Service Workloads**, there are three actions that can be taken:

- Stop
- Undeploy - The Undeploy action allows the platform owner to uninstall an instance of this service.
- Move - The Move action allows the platform owner to move the service to a different server. When moving a **Service Workload** you can let Apprenda choose a suitable server for you (based on role and resource allocation) or choose it yourself.

Move a Workload

Workload: Apprenda.SaaSGrid.TenantPortal.TenantPortalService

The workload will be moved to a new server. *This will result in a new process on the target server.*

Server: Let Apprenda Choose a Server

The platform will select a suitable server to run the new workload.

Let Me Choose a Server

Cancel

Deploy

Managing Version Components

The **Components** tab allows you to manage pieces of the application that can be deployed to the platform. For **Service** components you can configure **Service Level Options** for any new instances that get deployed in the future, by clicking on the arrow and selecting **Edit Service Level Options**.

Service Level Options

Autodeploy First Instance

Allow the fabric to proactively deploy one instance of this service if a request is made for the application and there are 0 running instances. If this happens, the request will pause until instance 1 is autodeployed and capable of responding, at which point the request will resume.

Garbage Collection Victim

Idle instances of this service should be eligible victims of fabric-wide service instance garbage collection. Instances are allowed to be shutdown by the fabric if no request reaches them for an extended period of time. This frees memory and disk space occupied by the shutdown instance.

Revivability and Location Affinity

Once an instance of this service starts, the fabric should make every attempt to restart it on the same server if it is stopped or crashes. Check this if the service has a dependency on underlying server capabilities (for instance, IIS or SQL Server).

Use Default Service Account Credentials (saasgridsystem@apprenda.local)

(Uncheck this to supply alternate credentials under which you would like this service to run.)

Browsing the Apprenda Environment's Repository

The **Repository Browser** page can be used to browse any disk location accessible on your network; however, it is designed primarily to allow you to use the SOC to access your environment's Repository, a centralized location (typically configured during installation as a network path or share) that Apprenda uses to manage both application and node file assets.

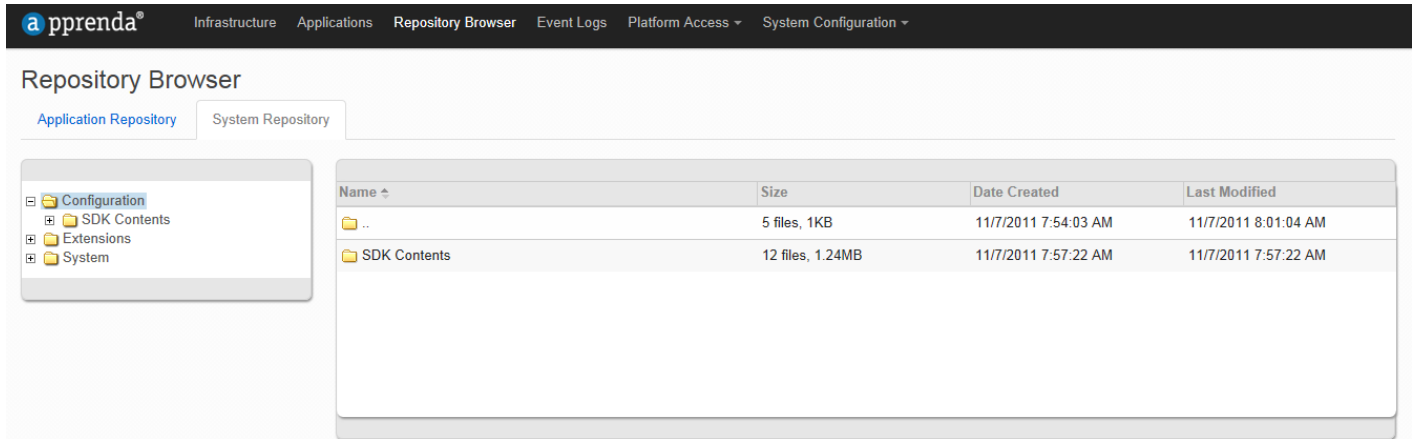
By default, the File System page will load the mount points for two essential components of the Repository:

- The Application Repository: a list of applications in the Repository.
- The System Repository Root: the part of the Repository dealing with core pieces of Apprenda.

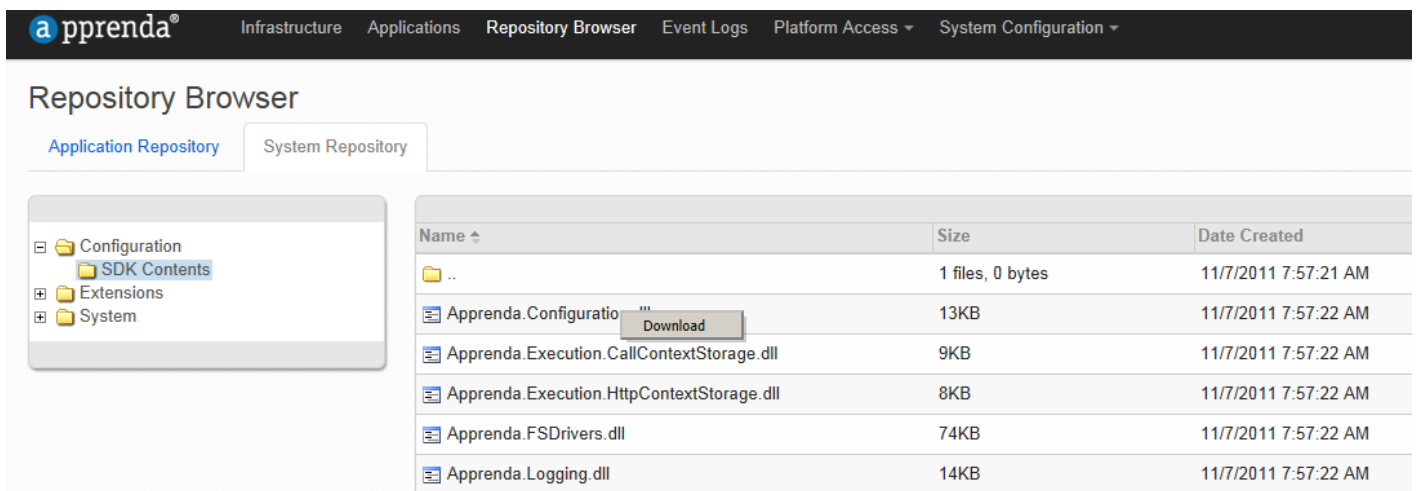
Navigating the Repository

To select one of the default Repository mount points, click on the appropriate tab. The mount point itself will appear in the left pane with its folders hierarchically structured as a tree. In the right pane you will see the first level of folders and files.

To browse files, double-click on a folder to display its contents in the right pane. Once you are in a folder or sub-folder, double-click on the ".." folder to move up one level of the file hierarchy:



To download a file, right click on it and select **Download**:



Managing the Platform Registry

Registry Settings are settings internal to Apprenda that are available to the entire system. These settings are not isolated to a particular customer or application but are instead intended to affect behavior throughout an entire Apprenda instance.

?There are two categories of Global Settings:

Core Settings are essential to Apprenda's operation and cannot be added or removed; most of these were set for you when you first installed Apprenda. Although the settings themselves cannot be removed, values for some settings can be changed once Apprenda is running.

Custom Settings are a type of setting that can be added to a live instance after it is up and running. They are intended as a means by which values can be set for functionality (such as business extension workflows) added to the platform. Future versions of the Apprenda runtime API will include the ability to query these custom settings from any user interface or service running on Apprenda, giving developers the ability to provide dynamic configuration to their deployed application components.

Modifying a Registry Setting

To modify a Registry Setting; click on the setting you wish to change and input the new value in the pop up window. Click **Save**.

Adding a Registry Setting

To add a new Registry Setting; click on the **Add a Registry Setting** button and input the new setting information. Click **Save**.

Deleting a Registry Setting

To delete an existing **Custom** setting; click on the setting you wish to delete and click **Remove**.

List of Registry Settings

The following list summarizes the Platform Registry settings...

Name	Explanation	Values
AccountPortalExtServices	Aliases of Account Portal extension services.	Comma separated values of alias/class name (app1/extclass, app2/extclass...)
AllowIdentityFederation	Allow identity federation for the environment.	True, False
AllowUiDeploymentToExceedCeiling	Allow the deployment of frontend workloads to exceed the ceiling.	True, False
Apprenda.CitadelEnableAutoComplete (valid as of version 3.1.2)	Allow autocomplete for password fields	True, False
Authentication.DefaultSessionDurationInMinutes (valid as of version 3.0.5)	Set a platform-wide inactive session duration.	Any positive integer
CitadelExtServices	Aliases of Citadel extension services.	Comma separated values of alias/class name (app1/extclass, app2/extclass...)
Citadel.Login.ForwardUrlOverride (valid as of version 3.0.3)	URL to which users are forwarded upon login to the Platform	Any URL valid for Platform users
Citadel.LogoutRedirectUrl (valid as of version 3.1.1)	URL to which users are forwarded upon Platform logout	Any URL valid for Platform users
Citadel.Signup.DevelopmentTeamSignupUrlOverride (valid as of version 3.0.3)	URL to which users are forwarded upon clicking on the "Create a Developer Account" link	Any URL valid for Platform users

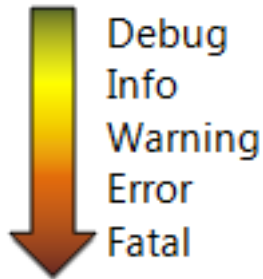
Citadel.Signup.EndUserSignupUrlOverride (valid as of version 3.0.3)	URL to which users are forwarded upon clicking on the "Create an End User Account" link	Any URL valid for Platform users
Citadel.Signup.ForwardUrlOverride	URL to which users are forwarded upon creating an Account for their Organization	Any URL valid for Platform users
DeveloperPortalExtServices	Aliases of Developer portal extension services.	Comma separated values of alias/class name (app1/extclass, app2/extclass...)
DeploymentSettings.ApplicationServices.Default	Default Apprenda services for guest applications.	None, Authentication, Authorization, Multitenancy, Billing
DeploymentSettings.ApplicationServices.MaximumApplicationServiceLevel	Maximum allowable Apprenda services for guest applications.	None, Authentication, Authorization, Multi-tenancy, Billing
DeploymentSettings.PersistenceDataModel.Default	Default database deployment model for guest applications.	Commingled, Isolated
DeploymentSettings.PipelineMode.Default (valid as of version 3.0.2)	Default pipeline mode for applications with no platform services.	Classic, Integrated
DeploymentSettings.PresentationDeploymentStrategy.Default	Default UI deployment model for guest applications.	IsolatedAppTenantRoot, IsolatedRootAppTenant, CommingledRootApp, CommingledAppRoot
LoadManagement.IndexOfFirstApprendaRule (valid as of version 3.1.2)	Specifies the starting index for Platform URL rewrite rules, allowing custom URL rewrite rules to execute first	Any integer greater to or equal to 0; number must correspond to the quantity of rules that should run prior to Platform rules
Persistence.ProviderPartition.MaxSize (valid as of version 3.0.3)	Maximum size allowed for a provider database partition (in MBs).	Any positive integer
Persistence.TenantPartition.MaxSize (valid as of version 3.0.3)	Maximum size allowed for a tenant database partition (in MBs).	Any positive integer
PhysicalHost.CpuThrottling.AllocationFactor	Allocation factor for CPU throttling.	number > 0 (1=100%)
PhysicalHost.CpuThrottling.TotalCpuThreshold	Percentage of total CPU needed to reach in order to start throttling guest apps.	0-100 (represents %)
PhysicalHost.MemoryThrottling.AllocationFactor	Allocation factor for RAM allocation.	number > 0 (1=100%)
PhysicalHost.ResourceAllocation.DefaultServicePolicyId	ID of the service resource policy assigned by default to guest applications.	GUID of a service resource policy
PhysicalHost.ResourceAllocation.DefaultUIPolicyId	ID of the Frontend resource policy assigned by default to guest applications	GUID of a Frontend resource policy
PhysicalHost.ResourceAllocation.MaxDeveloperCpuAllocation	Sets the total amount, in MHz, of CPU that a given developer's applications can collectively use on the platform. For this setting to have any effect, resource throttling must	-1 signifies no limit; otherwise any positive integer

	be enabled. The total is calculated by measuring allocations against resource policies assigned to the applications. Applications deployed against an unlimited resource policy do not count towards the total.	
PhysicalHost.ResourceAllocation.MaxDeveloperMemoryAllocation	Sets the total amount, in MB, of Memory that a given developer's applications can collectively use on the platform. For this setting to have any effect, resource throttling must be enabled. The total is calculated by measuring allocations against resource policies assigned to the applications. Applications deployed against an unlimited resource policy do not count towards the total.	-1 signifies no limit; otherwise any positive integer
PhysicalHost.ThrottlingEnabled	Enable throttling for the Apprenda instance	True, False
PhysicalHost.TrackUtilization	Track utilization data for the Apprenda instance.	True, False
PhysicalHost.UtilizationPollingIntervalSeconds	Interval between each polling for utilization data.	Time in seconds
PhysicalHost.UtilizationReportingInterval		Time in seconds
Presentation.Packing	Influences server selection when deploying a UI. When compressed (dense) UI's will deploy to machines that already have user interfaces if space is available. If balanced (sparse) they will deploy to a host with the fewest user interfaces. Applies only when resource throttling is enabled, and does not apply to UI's who have an resource policy with no limits for the corresponding packing type.	Sparse, Dense (default)
Presentation.PackingType	When deploying UI's, the determination of which server to use is done by measuring CPU usage or memory usage. See Presentation.Packing for more information.	Cpu, Memory(default)
Presentation.PathBasedUrlsEnabled	Enable path based URLs for guest applications.	True, False
ProviderSignUpStatus		Open
ResourcePolicies.MaxCpuCores	Maximum CPU cores available for resource policies.	Number of cores
ResourcePolicies.MaxCpuSpeed	Maximum CPU speed available for resource policies.	In MHz
ResourcePolicies.MaxMemory	Maximum RAM cores available for	In MB

	resource policies.	
ResourcePolicies.MinCpuCores	Minimum CPU cores available for resource policies.	Number of cores
ResourcePolicies.MinCpuSpeed	Minimum CPU speed available for resource policies.	In MHz
ResourcePolicies.MinMemory	Minimum RAM cores available for resource policies.	In MB
ResourcePolicies.StepCpuCores	Increment between CPU cores when creating resource policies.	Integer > 0
ResourcePolicies.StepCpuSpeed	Increment between CPU speeds when creating resource policies.	Integer > 0
ResourcePolicies.StepMemory	Increment between RAM allocations when creating resource policies.	Integer > 0
Services.Packing	Influences server selection when deploying a service. When compressed (dense), services will deploy to machines that already have services if space is available. If balanced (sparse) they will deploy to a host with the fewest services. Applies only when resource throttling is enabled, and does not apply to UI's who have an resource policy with no limits for the corresponding packing type. Note that services will only deploy to services hosting UI's and databases when there is no other server available for the deployment.	Sparse, Dense (default)
Services.PackingType	When deploying services, the determination of which server to use is done by measuring CPU usage or memory usage. See Services.Packing for more information.	Cpu, Memory (default)
StorageManager.SqlServerTarget.machine_name	Setting used for remote management of databases.	Name of remote SQL Server instance
StoreFrontExtServices	Aliases of StoreFront extension services.	Comma separated values of alias/class name (app1/extservice, app2/extservice2...)
TenantSignUpStatus		Open
UiDeploymentCeiling	Maximum frontend workloads that can be deployed to each frontend server.	Integer > 0
UiDeploymentThreshold	Number of frontend workloads that need to be reached in order to automatically deploy to other frontend servers.	Integer > 0
UserInterfaceDeploymentLevel	Determines the deployment model used for UIs.	

Managing the System Event Logs

Any web service or user interface hosted on one of your Apprenda servers can contact the system logging service using the Apprenda API. The logging service will only allow events to be logged at the current Global Log Level or higher, in order of severity, where severity is measured as follows (from low to high):



For example, if the Global Log Level is set to Warning, the logging service will accept and display log entries for Warning, Error, and Fatal events. Based on expected usage of the logging service, the Debug level will result in the highest number of event log entries across the system.

Viewing a Log Entry

The **Event Logs** list shows a paged list of events that have been captured by the logging system in the past 24 hours. The default view displays the Level, Date and Time, Source, Machine (server), and a truncated view of the log entry text for each entry. Use the column headers to sort by any column. You can also search for source, machine or text via the **Search** textbox and/or restrict by timestamps.

Level	Timestamp	Source	Machine	Text
INFO	11/15/2011 10:19:47 AM	Apprenda.SaaSGrid.SMART.LocalDep	ERIK-VM2	UndeployService() '42d0b567-f78d-424c-ba89-f034f8427427' was undeployed successfully.
INFO	11/15/2011 10:19:47 AM	Apprenda.SaaSGrid.SMART.LocalDep	ERIK-VM2	Erase(): Erasing launchpad 'ApprendaPlatform\Container\LaunchF-f78d-424c-ba89-f034f8427427'
INFO	11/15/2011 10:19:47 AM	Apprenda.SaaSGrid.SMART.Bootstrap	ERIK-VM2	OnClosing(): Completed OnClosing for "HelloApprenda.HelloService (Id=42d0b567-f78d-424c-ba89-f034f8427427)"
INFO	11/15/2011 10:19:47 AM	Apprenda.SaaSGrid.SMART.Bootstrap	ERIK-VM2	OnClosing(): Entered for instance 'HelloApprenda.HelloService (Id=42d0b567-f78d-424c-ba89-f034f8427427)'

Viewing 1 - 4 of 4

Note: The Event Logs view will display by default the last 24 hours' worth of logs. You can use the Start Date and End Date textboxes to search further than the past day.

In order to view a specific log message, double click on its row to open a pop up containing the full log text.

11/15/2011 10:19:47 AM
✕

[Open in new window](#)

INFO event created by **Apprenda.SaaSGrid.SMART.LocalDeployment.LaunchPad** from machine ERIK-VM2.

```
Erase(): Erasing launchpad 'ApprendaPlatform\Container\LaunchPads\42d0b567-f78d-424c-ba89-f034f8427427'
```

Deleting Individual Log Entries

You can delete a specific log entry from the logging system by selecting it out of the list (use **Ctrl + Click** to select multiple entries). Click **Delete Selected** in the left toolbar to delete selected log entries.

Logging System Context Awareness

Every event explicitly logged using Apprenda’s logging API maintains contextual awareness of the prevailing Apprenda contexts present at the time that event is captured. In addition to the timestamp of the event, every log entry in the Event Logs can contain:

Application: Developer Portal Version 1 316b32db-e812-40b8-9be0-9a4292bc95cc

SessionId: 9c7765d9-feef-4989-8cf6-eb175c4bfd12

SessionActive: false

ProviderName: Apprenda, Inc 00000000-0000-4000-0000-000000000001

Tenant: CS Test f487ac9b-1bc5-4385-b7e3-ab9aff77edfd

User: elustgarten@apprenda.com 6ea8e5fd-6db0-4a84-9bc1-9f9400854fc4

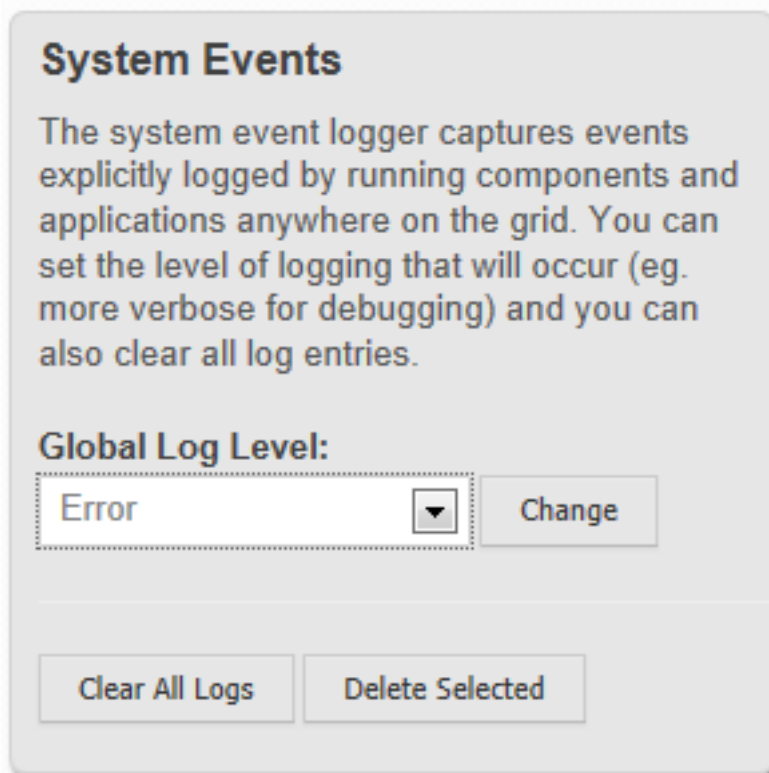
- **Application** – The application and version that logged the event.
- **Session ID** - The unique ID of the SaaSGrid session during which the event occurred.
- **SessionActive** - Flag specifying if the session is still active.

- **ProviderName** -The name and unique ID of the provider that deployed the application that logged the event.
- **Tenant** -The name and unique ID of the organization who owns the user and session during which the event occurred.
- **User** -The email address and unique ID of the user who made the request during which the event occurred.

Contextual awareness is extremely useful in determining the details around a particular event. For instance, finding contextual patterns in recurring events can be used to determine whether or not the event pertains only to a particular customer or user.

Setting the Global Log Level

To set the global log level, which is the level at which the logging system will accept log entries from any entity, select the desired level from the drop down list in the left panel, and then click the **Change** button. The logging system will immediately begin to accept and display log entries at the new global level.



Clearing the Logs

Periodically, you may wish to empty the system store for logged events. To do so, click the **Clear All Logs** button.

Note: Clearing the logs does not perform any sort of backup prior to removing log entries. If you wish to store log entries for historical purposes, you should employ a regular database backup strategy for the Apprenda core databases.

Masking the Logs

Some log entries may contain sensitive data that you do not want to be stored within the logging service's database. If one or more Log Masks are specified, Apprenda will perform a search & replace action on the text of all log entries as

they are processed by the logging system.

To create a new Log Mask, enter the text that Apprenda should search for within log entry text along with the text that you want Apprenda to replace it with. For example, if you want to mask the known password text “mypassword,” you might enter something like this:

Log Masks

Use log masks to replace content in log entries before they are saved in the logs. For example, replace 'password' with '*****'.

Search Pattern

Replacement Text

Is Regular Expression

Apprenda’s logging system will replace the string “mypassword” with the string “*****” within the text of any event sent to the system at any logging level. It is important to note that this action occurs prior to the log entry actually being stored in the logging system’s database to ensure that the data you are trying to mask is not persisted. Click the **Create Mask** button to create your Log Mask and immediately apply the mask to incoming log events. Apprenda will not retroactively perform masking on events already in the logging system’s database.

Overriding the Logs

Setting the Global Log Level to a less severe level (Debug, Information) can result in a large number of events being sent to the logging system. Typically, software developers will use the Debug and Information log levels within application code to capture execution information during normal operation of the software that can be useful if something goes wrong. Apprenda’s core services work this way. Therefore, setting the Global Log Level to a less severe level can result in a lot of “noise” in the logging system from core services and deployed applications. For example, when the Global Log Level is set to Debug, all Apprenda routers on all machines will log information about the messages that they are routing between services. Apprenda allows you to create a **log override**, which enables you to capture events at a certain level for specific applications while the rest of the system remains at the Global Log Level.

Click the **Overrides** tab to view a list of the log overrides currently in place. To create a new log override, enter the unique Version ID for the application version whose log level you want to override. The unique ID is located in the Version Information page for the application:

Version
 Version 1 (v1)
 40fbb645-f07f-4f59-a4eb-4a692452f709
 Published

Application
 Account Portal
 account

Developer
 Apprenda, Inc

Platform Services

- Authentication
- Authorization
- Multi-tenancy

Next, select the log level for this override.

You may filter your log override with additional criteria for finer grained control over the log entries that will be captured by this override. One of the following filter types may applied per log override:

- Tag (Code Source) - A string that matches the source tag for the log entry.
- Organization (Tenant) - The unique ID of the organization entity within Apprenda that is the prevailing context at the time the log entry is created.
- Subscription (User) - The unique ID of the user entity within Apprenda that is the prevailing context at the time the log entry is created.
- Message (Message Text) - A string that matches the text of the logged message.

Additionally, you may provide a semi-colon or space delimited list of email addresses to notify when a log entry that matches your new log override criteria is captured. The logging system will automatically dispatch emails to the addresses you provide here in addition to capturing the event in the logging system.

The following settings would result in Apprenda capturing all Debug events that occur while Organization D85C315F-0DE4-4BA7-8128-B5BDB5DB272A is using application version 23f78324-5c8f-4a53-8609-f5860bb2f1b0, and event-notifications@contoso.com will receive an email from the system:

Log Overrides

Use log overrides to customize the log levels for a specific application or application/organization/user combination. Choose a version and a log level for the override and then add a filter to further limit your results.

VersionId:

Level:

Additional Criteria

Filter:

Filter Type:

Notify

Email Recipients

Enter email addresses separated by ';' or whitespace (space, tab, new line)

Editing a Log Override

Select a log override in the list of current overrides and then click the **Edit Selected Override** button. The parameters of the selected override will be displayed in the **Log Overrides** form in the left panel. From here you can make necessary changes to your override. Click the **Save Override** button to save those changes. The action will be applied immediately, but will not affect any log entries that have already been captured by the previous override settings.

Removing a Log Override

Select a log override entry in the list of current overrides (Ctrl + Click to select multiple entries) and then click the **Delete Selected Overrides** button to remove the override(s). This action will be applied immediately, but will not remove any log entries that have already been captured by deleted overrides.

Configuring Resource Policies and Throttling

Resource Throttling on the Apprenda platform is designed to give Platform Administrators the ability to limit the resources a given guest Application can consume. In addition, a Platform Administrator can set **Resource limits per Developer group**, thus ensuring that no one group monopolizes resources on the Platform. Both forms of Resource Throttling are explained in greater detail below.

Please note that [Resource Utilization Tracking](#) is governed by separate controls that are explained [here](#).

Configuring Resource Policies for Guest Applications

Navigate to the **Resource Policy Management** page in the SOC (via the **System Configuration** link in the top menu bar).

apprenda®

[Infrastructure](#)
[Applications](#)
[Repository Browser](#)
[Event Logs](#)
[Platform Access](#)
[System Configuration](#)

Resource Policy Management

Resource Policies

These policies will allow the system to govern the consumption of CPU and Memory by application components on the platform.

[Enable Throttling](#)

Defaults

The chosen default policies will be applied to components without policies when enabling throttling. They will also be used as the defaults if you choose to enable Automatic Policy Assignment.

Automatic Policy Assignment:
 Enabled

Service Default:

User Interface Default:

Distribution Strategies

You can choose how you would like the platform to distribute services and UIs across machines. A Balanced strategy will favor distribution over utilization while a Compressed strategy will attempt to favor utilization over distribution.

Services Strategy:

User Interfaces Strategy:

[Save Changes](#)

Policy Management
[Workload Allocations](#)
[Component Assignments](#)

[New Policy](#)
Filter:
Show: Inactive Undeployable

Default

This policy is the default policy that will be applied to all applications that do not have a policy selected.

CPU Speed: Max available across 1 core
Maximum RAM: Max available
Active: Yes
Deployable: Yes

[Edit](#) [Delete](#)

Policy with Large Limits

No description

CPU Speed: 2000 MHz across 2 cores
Maximum RAM: 2000 MB
Active: Yes
Deployable: Yes
Cost: \$0.25 per instance hour

[Edit](#) [Delete](#)

Policy with Small Limits

No description

CPU Speed: 500 MHz across 1 core
Maximum RAM: 500 MB
Active: Yes
Deployable: Yes
Cost: \$0.07 per instance hour

[Edit](#) [Delete](#)

If any active, deployable Resource Policies have already been configured for your Platform, you will see them listed on the right side of the screen in the **Policy Management** tab. By default the Apprenda platform is installed with an unlimited default Resource Policy. You may use this if you wish, or configure your own policies for Guest Applications.

To configure a new Resource Policy, click on the **New Policy** button at the top of the page. This will open a Resource Policy Definition window:

Resource Policy Definition

Applies To: All Component Types ▼

Name:

Description:

A short description of the policy for the developers.

Notes:

Notes will never be shown to the developers.

Active:

Deployable:

CPU Speed (MHz):

Number of Cores:

Maximum RAM (MB):

Cost Definition:

When creating a policy, you can make it available for Application **Services**, Application **User Interfaces**, or **both**; this allows you create separate resource limits for different Application components. After you input a name for the policy, you can also add a **Description** (which can be seen by Developer groups when selecting a policy) as well as any **Notes** (which are visible to Platform Administrators through the SOC but are not visible to Developer groups).

You can then mark the policy as **Active** and **Deployable**.

When a Resource policy is **Active**, it is available for Developer groups to assign to Application components. When Resource Throttling is enabled, an Application's Service and UI components must have Active Resource policies assigned; otherwise, the Application cannot be promoted to the Sandbox or Published stages, and components cannot

be deployed through the Cloud Control panel of the Developer Portal. An Inactive policy, however, will not prevent components from being deployed when an Application is launched, and a Platform Administrator can deploy its components through the SOC.

When a Resource policy is **Deployable**, components tied to it can be deployed by launching the Application, even if the policy is Inactive. Applications with Undeployable components cannot be launched; an Application with UI components assigned to Undeployable components cannot be promoted to the Sandbox or Published stages.

To set the Resource limits, slide the CPU speed, Number of Cores, and Memory bars to the desired limits (it is possible to set an unlimited amount of CPU speed and/or memory). The minimum and maximum limits available, as well as the increments in between, are configurable by changing the following [Platform Registry Settings](#):

Name	Explanation	Values
ResourcePolicies.MaxCpuCores	Maximum CPU cores available for resource policies	Number of cores
ResourcePolicies.MaxCpuSpeed	Maximum CPU speed available for resource policies	In MHz
ResourcePolicies.MaxMemory	Maximum RAM cores available for resource policies	In MB
ResourcePolicies.MinCpuCores	Minimum CPU cores available for resource policies	Number of cores
ResourcePolicies.MinCpuSpeed	Minimum CPU speed available for resource policies	In MHz
ResourcePolicies.MinMemory	Minimum RAM cores available for resource policies	In MB
ResourcePolicies.StepCpuCores	Increment between CPU cores when creating resource policies	Integer > 0
ResourcePolicies.StepCpuSpeed	Increment between CPU speeds when creating resource policies	Integer > 0
ResourcePolicies.StepMemory	Increment between RAM allocations when creating resource policies	Integer > 0

Please note that choosing "unlimited" for a resource type (Memory, for instance) will mean that any component tied to that Resource Policy will not be throttled based on that resource type.

Finally, you can opt to denote any Cost Definition involved for resources used. This will be visible to Development groups, but will appear for communication purposes only, as Apprenda does not automatically calculate or regulate chargeback for Developer group resource usage.

Once you have configured your Resource Policy, hit the **Save** button. Please note that once you have saved a policy, you will only be able to make cosmetic changes, such as updating the name or description, through the **Edit** function. You will not be able to update policy limits (but can, of course, create another new policy). You can **Delete** a policy by clicking on the corresponding **Delete** link.

To set a policy as the Service or User Interface default, simply select it in the pulldown menu appropriate section on the left side of the page, then click **Save Changes**; you can choose different policies for Services and UIs. The default Resource Policy is (if active), available for all components of its type, and Developer groups will be given the option of assigning it to their Application's components through a special link. It will not be automatically assigned to new components, however, unless **Automatic Policy Assignment** is enabled (which can be done by checking the box just above the default Policy assignment controls). If the default Resource Policy is not configured with Automatic Policy Assignment enabled, Developer groups will have to manually select a policy before deploying their applications.

Tips for Resource Policy Management:

It is highly recommended that Platform Administrators clearly communicate any anticipated changes in Resource Policy availability to Developer groups. When phasing out a policy, for example, new policies should be made available and clear deadlines for unassigning retiring policies should be set. As needed, you can also use the Active and Deployable flags to help enforce such deadlines. Removing the Active flag, for instance, will prevent any further Developer groups from assigning the policy, and will encourage those that have it assigned to adopt a new policy, as their ability to manipulate the Application will be restricted. You can then remove the Deployable flag, which will prevent launching of the Application, as another step of phasing out the old policy and encouraging the adoption of a new one.

Enabling Resource Throttling

Once you have configured active, deployable Resource Policies for guest Applications, you can enable Resource Throttling (it is possible, but not recommended, to enable throttling without first configuring Resource policies; this will, however, prevent Developer groups from deploying or launching their Applications). To do this, simply click on the **Enable Throttling** button on the upper left side of the of the **Resource Policy Management** page. Once enabled, a **Disable Throttling** button will appear that will allow you to disable Resource Throttling.

Once enabled, Resource Throttling works by limiting the Resources that a guest Application's components can consume to those defined in their Resource Policy. It also determines the number of "slots" available on the platform to host Service and UI deployments tied to each Resource Policy with limited (and not "unlimited") resource definitions. In other words, once a Service or UI is deployed, a section of CPU utilization and Memory equal to the limits set forth in its Resource Policy are "reserved" for that Service or UI Deployment. Once the "reserved" resources reaches the total resources available on the platform, no new deployments will be allowed.

Because it is unlikely that all guest Applications hosted on the Platform will require all of their resource allocations at once, the **PhysicalHost.CpuThrottling.AllocationFactor** and **PhysicalHost.MemoryThrottling.AllocationFactor** settings can be configured to tell the Platform to allow the "reserved" resource allocations to exceed the actual physical resources present. For each of these settings, a value of **1** will result in a strict correlation of the "reserved" resources to the platform's actual resources; a value of **2** will allow the "reserved" resource allocation a limit of twice the actual platform resources; a value of **three** will allow the "reserved" resource allocation a limit of three times the actual platform resources, and so on.

Please note that there is also a minimum CPU usage on the platform that must be met before Throttling will take effect. This is controlled by the `PhysicalHost.CpuThrottling.TotalCpuThreshold` setting. The default value for this setting is 75, meaning that CPU usage must hit 75% before Throttling takes effect (although once it take effect, it will not stop if CPU usage falls below this number). This setting, along with those mentioned above, can be configured on the [Platform Registry Settings](#) page.

Distribution Strategies

Once Resource Throttling is enabled, you can configure how Apprenda deploys Service and UI components for guest Applications. At the bottom left of the Resource Policy Management page, you will see the **Distribution Strategies** section, which includes a pull-down menu each for Service and UI deployments. For each of these, you can decide to determine future component deployment according to either RAM or CPU usage. You then must select a **Balanced** or **Compressed** strategy. If **Balanced** is selected, the platform will attempt to distribute component deployment evenly among the available servers for that component type (UI deployments, you'll recall, can only be deployed to servers equipped to handle UI deployments). If **Compressed** is selected, the platform will attempt to fill up a server before it begins to deploy to a different server.

Setting Resource Limits for Developer Groups

Before Resource Limits can be set for Developer groups (where a "Developer group" is defined by Apprenda platform Developer Account), at least one active Resource Policy must be configured and Resource Throttling must be

enabled. This is because Developer group Limits are enforced on an Application component deployment basis as determined by the Resource Policies attached to a Developer group's Applications. This means that Apprenda determines when a Developer group has met its limit by adding up all of the resource slots that are "reserved" by the group's deployed Applications, and these "reserved" resource slots are determined by guest Application Resource Policies. Once a Developer group has met its limit, no further Service or UI deployments are allowed from that Developer group's Applications. The group may, however, choose to undeploy instances in order to free up space in their group allocation.

Resource limits for Developer groups can be configured through the following [Platform Registry Settings](#):

Name	Explanation	Values
PhysicalHost.ResourceAllocation.MaxDeveloperCpuAllocation	Sets the total amount, in MHz, of CPU that a given developer's applications can collectively use on the platform. For this setting to have any effect, resource throttling must be enabled. The total is calculated by measuring allocations against resource policies assigned to the applications. Applications deployed against an unlimited resource policy do not count towards the total.	-1 signifies no limit; otherwise any positive integer
PhysicalHost.ResourceAllocation.MaxDeveloperMemoryAllocation	Sets the total amount, in MB, of Memory that a given developer's applications can collectively use on the platform. For this setting to have any effect, resource throttling must be enabled. The total is calculated by measuring allocations against resource policies assigned to the applications. Applications deployed against an unlimited resource policy do not count towards the total.	-1 signifies no limit; otherwise any positive integer

Please note that no throttling will take place when these values are set to **-1**; also, the total resource allocation for a Developer group will not take into account any Application components deployed under and "unlimited" Resource Policy.

Resource Policies Required by Apprenda

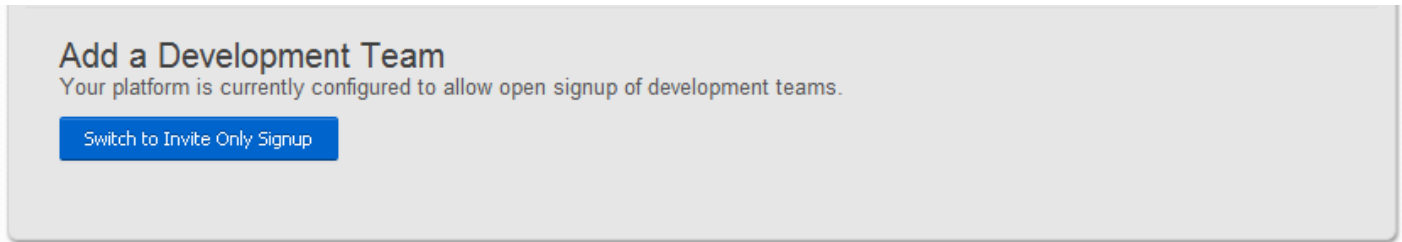
Your Platform was installed with two Resource Policies: an **Apprenda Core Services** policy and a **Cataloging Service** policy. Both were installed as **Inactive**, meaning that they are not available for use for guest Applications. These policies are essential for your Platform to work properly; as such, they should not be altered.

Controlling Platform Access

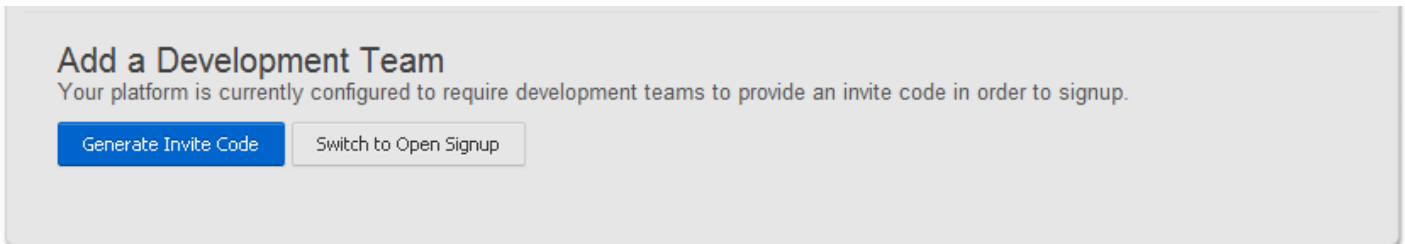
Managing Development Teams

By default, the Apprenda platform allows any development team to register for access simply by entering some basic account information. If you'd prefer to have more control over which development teams are allowed to use the system, you can switch from an **Open Signup** mode to an **Invite Only Signup** mode, where a development team must provide an invite code before being allowed to register.

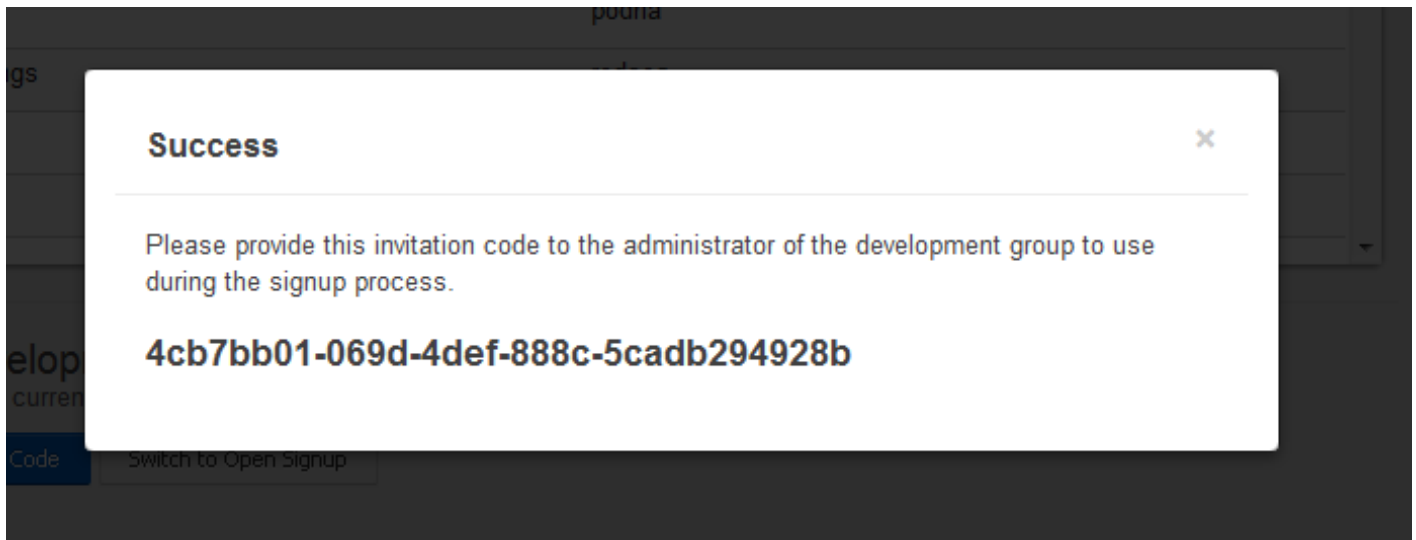
To toggle the signup mode, select the **Development Teams** option under the **Platform Access** menu and click the **Switch to Invite Only Signup** or **Switch to Open Signup** button.



If your platform is configured to require an invite code prior to registration, you'll need to use the **Generate Invite Code** button to create a single-use token.



Once the invite code has been generated, you'll want to provide the token to the development group for entry at registration time.



Disaster Recovery Planning

Objective

The objective of this document is to provide a framework of issues and concerns associated with recovering Apprenda in the event of a disaster. To help understand Apprenda recovery mechanics, it is important to first understand Apprenda's fundamental purpose and architecture. This document describes Apprenda's purpose and core structure to the extent that is necessary for disaster recovery planning.

Summary

Apprenda is a container technology that acts as a runtime and operating toolset sitting above network infrastructure but beneath "guest applications" that it hosts. This document will frequently refer to "guest applications" and Apprenda, where Apprenda is used to describe systems components that define Apprenda sans the presence of "guest applications."

Topologically, Apprenda is a fabric of service oriented components distributed amongst a collection of networked servers. The topology of this fabric is ever-changing as Apprenda constantly performs actions to facilitate the deployment, execution, and service level fulfillment of guest application services, web sites, and databases. Therefore, Apprenda treats a majority of these components as volatile resources that could, at any time, be redistributed across this network of servers. Being a middleware technology by definition, Apprenda orchestrates a collection of technologies below it to perform many operational actions on behalf of applications above it. Because of this architecture, most disaster recovery plans that involve Apprenda will actually be centered on standard recovery procedures for underlying technologies.

Apprenda's disaster recovery procedures are predominantly the same as any standard practices for an application hosting environment that employs Windows Server, Internet Information Services (IIS), and SQL Server technologies. That said, there are Apprenda-specific resources that should be part of any complete disaster recovery plan, and this document describes them by where they exist in the aforementioned topology. Specifically, this document addresses:

- How Apprenda relies on certain components and their configuration, and what it means to replicate those components to another physical environment.
- Which Apprenda assets need to be replicated and which assets are optional.
- The Apprenda configuration, if any, that must change between an active environment and an environment deployed in a recovery scenario.

It should be noted that Apprenda maintains its own inventory of data that allows for various portions of a Apprenda instance or a guest application to be rebuilt without having replicated the full environment. This document will outline backup strategies that focus on ensuring the availability of this inventory or on replicating environment assets so that a rebuild is not required. Regardless of approach, a high level overview of a recovery scenario is as follows:

1. Insure Apprenda management tools are available in a secondary recovery environment.
2. Insure key components are replicated and/or available as backups in a secondary recovery environment.
3. Restore backed up databases and files.
4. Make changes, if necessary, in key configuration files and database tables. A virtual environment with fully equivalent URN structures may obviate the need to make configuration changes and is recommended as a target deployment mechanism.
5. Start Apprenda nodes.

Backup strategies, where suitable, may take on either active/active or active/passive disaster recovery models.

The Core Apprenda Databases

During installation, the Apprenda Installation Wizard provisions Apprenda's core databases. These databases are used by Apprenda's fabric services during normal operations. The core Apprenda databases are:

- Apprenda Core
- Apprenda Authentication
- Apprenda Billing

- 40fbb645-f07f-4f59-a4eb-4a692452f709
- 316b32db-e812-40b8-9be0-9a4292bc95cc

These databases should be part of a regular backup strategy for the purposes of disaster recovery as well.

The Apprenda Repository (Recovering Application Binaries)

Apprenda stores all fabric service binaries, as well as guest application binaries, in a central location known internally as the Repository. The Repository is a Windows share created during installation that exposes a well known location for application binaries to the rest of the fabric. This location is used, for example, during deployment of a guest application's user interface(s) - Apprenda copies the user interface binaries from the Repository to a web server prior to configuring an IIS website to utilize those binaries.

Because the Repository holds these files, it is an imperative part of normal Apprenda operations and must be accessible in order to facilitate many of Apprenda's actions. From a disaster recovery standpoint, this location on disk should be backed up regularly. We recommend employing a backup tool that is capable of monitoring disk activity or file property changes, allowing a backup to trigger in the event that the contents of the Repository change.

The Web Tier (Recovering the Presentation Model)

Apprenda provisions guest application user interfaces (ASP.NET, HTML, etc.) on servers by configuring websites in Internet Information Services (IIS). The process includes copying guest application binaries to a disk location on the IIS server, configuring websites in IIS whose base directories are that disk location, and configuring website bindings. The process also includes registering configuration information with Apprenda's global Load Manager service, which routes inbound HTTP traffic to the appropriately configured website based on request characteristics (Is the user authenticated? Which customer does the user come from?). This Load Manager service exists on your primary web server (typically the first web server you added to your Apprenda fabric).

Internally, Apprenda refers to the makeup of web servers and deployed application user interface pieces across the fabric as its Presentation Model, and maintains a persisted representation of this model at all times. Using this model, it is possible to "playback" provisioning events and effectively reconstruct all previously deployed guest application user interfaces. Apprenda provides the tools to do so. In the event that part, or all, of the web tier of your Apprenda fabric should crash, the entire model can be rebuilt on new servers in an automated fashion.

If more granular disaster recovery is desired, we recommend following standard backup procedures for IIS servers as specified by Microsoft. In this case, our recommendation is to perform routine backups of your web servers' applicationHost.config file (metadata information about IIS websites), and c:\Apprenda\SiteData and c:\Apprenda\PublicSiteData, which contain the binaries that have been provisioned by Apprenda for all guest application user interfaces. Using these backups, you can restore a single web server without Apprenda's intervention.

The Services Tier (Recovering Deployed Services)

Apprenda's business services and message routing bus are comprised mainly of Windows Communication

Foundation services. Likewise, guest applications may use WCF services to encapsulate business logic. Apprenda treats all WCF services deployed on the fabric as volatile instances. When a WCF service is deployed on the fabric, a server is chosen and the binaries for that WCF service are sent to it. The server then uses Apprenda's own services container to host an instance of that WCF service, exposing the WCF service endpoints to the rest of the fabric. There can be 0 to n individual instances of any WCF service on the fabric at any given time, and because they are isolated service instances, there can even be more than one on a single server. Apprenda's default routing strategy routes application messages across the fabric to known service endpoints irrespective of their physical location. The compute power behind a given application can be loosely correlated to the number of instances of its services that are running on the fabric (e.g. the number of instances that can be doing work for the application simultaneously). From the disaster recovery standpoint, this dynamic fabric architecture means that portions of the infrastructure can crash, and Apprenda will route application messages to remaining service endpoints. If the infrastructure crashes to the extent that there are 0 remaining instances of a given service, Apprenda will deploy instance #1 to any remaining available servers in order to fulfill application requests.

Because there is no set topological requirement for this services tier, the entire tier can be reconstructed simply by letting Apprenda redeploy service instances across the fabric. If a server crashes, a replacement can be joined to the fabric and brand new service instances deployed to it within a matter of minutes. In the meantime, the remainder of the fabric will be used to shoulder the services load of the missing server.

The Database Tier (Recovering the Persistence Model)

Similar to the web tier, Apprenda provisions guest application databases in SQL Server by executing guest application-provided SQL scripts. The primary type of SQL script that an application may provide is called an "application provisioning script". This script contains the SQL statements necessary to construct the application's database schema. Because Apprenda provides multiple data isolation policies, the resulting physical databases resulting from a single application provisioning script will differ based on the application's configured isolation policy:

Commingled Data

When an application is configured for the commingled data policy, Apprenda will provision shared databases for that application. Apprenda's commingling process transforms the database schema in a way that facilitates access and data storage by multiple customers to a single physical database.

Isolated Data

When an application is configured for the isolated data policy, Apprenda will provision a physical database per customer. Apprenda will not transform the application's data schema to facilitate commingling. Instead, there will simply be more physical databases for the application, each providing storage for a single customer.

Given these data isolation policies, and the fact that an application provider may manually slice customer data to facilitate "mixed mode" isolation beyond the application's configured policy, the database topology for a Apprenda environment can take on innumerable variations.

Therefore, as a general rule for disaster recovery, we recommend routine SQL server backups. This includes backups of physical databases as well as SQL Server configuration for those databases (logins, users, etc.). The SQL Server engine itself provides these backup mechanisms, as do many 3rd party backup applications. Although Apprenda

maintains a model representation of deployed guest application databases, it does not include stored application data in that model. Therefore, while the structural topology of guest application databases can be reconstructed by Apprenda alone, the data stored in those databases cannot. For this reason, Apprenda defers backup and restore responsibility to the capabilities of the SQL Server technology, and the onus is on the operator of the environment to perform regular data backups.

Conclusion

Apprenda's goal is to remain unobtrusive to standard practices for operating underlying technologies. As such, there are few additional requirements for disaster recovery when Apprenda is involved as the middleware technology. Standard practices for backup and recovery of disk, IIS configuration, and SQL Server configuration are sufficient for the recovery of a Apprenda environment. In summary, the specific disaster recovery touch points for Apprenda are:

- **Web Tier**
 - The c:\Apprenda\SiteData and c:\Apprenda\PublicSiteData folders on any web server participating in the Apprenda fabric.
 - The applicationHosts.config file, which contains IIS metadata for each web server.
- **Data Tier**
 - SQL Server backups of application database schema information, data, and backing database files.
 - SQL Server server-level configuration such as server logins, roles, etc of any database servers participating in the Apprenda fabric.
 - SQL Server backups of the core Apprenda databases as described in this document.
- **The Apprenda Repository**
 - Disk backup of the Apprenda Repository shares.

Server Roles

Each server of your Apprenda instance can take on varying hosting responsibilities. These responsibilities are determined by

1. The underlying software on the server (e.g. SQL Server).
2. The options you choose during installation.

The most basic role that a server can fill is to host *Services*. Additional roles include *Storage* and *User Interfaces*.

Services

Apprenda's most core architectural quality is that of a web services fabric. All servers on Apprenda are of the *Service* role and are capable of hosting WCF web services.

The high-level functionality that Apprenda provides for your web services include:

- Linear scale-out and load distribution across multiple servers
- High availability with built-in service revival and displaced service recovery
- New instance deployment via the Apprenda Operations Center and Provider Portal
- Automated teardown of unused services
- Protocol bridging services (e.g., bridging HTTP based requests to net.tcp)

During installation, the following two *Windows Services* will be installed on the server to support deployment:

- Physical Host Controller
- Service Container

Apprenda relies on its own WCF web service hosting capabilities for some of its functionality. One such service is the *Router*, which is a WCF service that is required on all machines participating in the grid and which helps services contact other services in the grid. Other Apprenda business services such as the *Billing Service* and several others must be deployed onto the grid (and oftentimes automatically re-deploy themselves based on the aforementioned high availability functionality discussed above).

User Interfaces

Perhaps no server is as important to end-users as the web server, as it hosts the web application with which they will interact.

The high-level functionality that Apprenda provides for your web applications include:

- On-demand provisioning of web sites for your tenant
- Full integration with Apprenda's patching mechanics
- Certificate management and SSL support
- Native load distribution across multiple servers

The fabric denotes the server as serving the *User Interfaces* role if the *User Interface Manager WCF Service* is running on the machine.

A supported version of Internet Information Services (IIS) must be installed and configured correctly in order for the *User Interface Manager* service to provision and manage web sites. During installation, support will be verified and the server will be configured appropriately.

When a request to deploy a web application is made, the fabric will select at random a server with the *User Interfaces* role, and deploy the application to it.

One *User Interfaces* server is designated as the *primary web server*. This server is configured to respond to all incoming web requests. Apprenda uses *Application Request Routing (ARR)*, an add-on to IIS, to route requests to the web server that actually hosts your application's user interface. The *Load Management Windows* service must also be running on the primary web server.

On a multi-node grid, servers that are the *User Interface* role are denoted as servers that are ineligible to accept additional web services by default. This means when Apprenda automatically deploys web services, Apprenda will not choose this server unless no other server is available. It is always possible, however, to deploy a service manually using the *System Operations Center* or *Developer Portal*.

Storage

Given the importance of application data to most applications, Apprenda provides powerful database deployment and hosting capabilities.

The high-level functionality that Apprenda provides for your databases include:

- Support for various data isolation models
- Scale-out across multiple servers with advanced data management capabilities
- Transactional deployment and patching mechanics to ensure data integrity and availability

A server is denoted as serving the *Storage* role if the *Storage Manager* WCF Service is running on the machine.

A supported version of SQL Server must be installed and configured correctly on the server in order for the Storage Manager service to function properly. During installation, support will be verified and the server will be configured appropriately.

On a multi-node grid, servers that are the Storage role are denoted as servers that are ineligible to accept additional web services by default. That means that when Apprenda automatically deploys web services, Apprenda will not choose this server unless no other server is available. It is always possible, however, to deploy a service manually using the *System Operations Center* or *Developer Portal*.

Resource Utilization Tracking

The information refers to the Apprenda platform's Resource Utilization Tracking capabilities. By default at installation time this capability is turned off. In order to activate Resource Utilization tracking, you must navigate to the Platform Registry page and set the **PhysicalHost.TrackUtilization** Registry Setting to **True**.

Please note that Resource Utilization Tracking is not automatically enabled when **Resource Throttling** is enabled. Please see [here](#) for information on [Resource Throttling and Resource Policies](#).

Background

Resource allocation and utilization tracking on the Apprenda platform is designed to give Platform Administrators the real-time knowledge they need to make informed capacity-planning decisions based on actual resource utilization as consumed by guest applications within the structure of resource policies offered to Developer groups. Likewise, the same metrics as seen through the lens of the Developer groups enable Developers to track and make better use of the resources they consume. In the end, the data offered by the Apprenda platform allows the Platform Administrator and the Developer groups to jointly maximize efficient use of the platform infrastructure.

Scope: Platform

Platform-wide metrics assist the Platform Administrator with such things as capacity planning, resource policy creation (resource slicing), and troubleshooting across all applications and Developer groups. For example, the

Platform Administrator may look at data indicating that there are a large number of sandboxed applications, and in anticipation of those application being published by Developer groups, the operator may add resources to the platform infrastructure. Likewise, the Platform Administrator may use resource policy assignment / actual utilization differentials to recognize that applications deployed to the platform can operate within smaller resource slices. Armed with this information, the Platform Administrator can adjust the resource allocation policies offered to Developer groups.

Published application versions

- Total Published application versions
- Published applications per Developer group
- Published frontend components (allocation, potential utilization)
- Published web service components (allocation, potential utilization)
- Active frontend workloads (utilization)
- Active web service workloads (utilization)

Sandboxed application versions

- Total Sandboxed application versions
- Sandboxed applications per Developer group
- Sandboxed frontend components (allocation, potential utilization)
- Sandboxed web service components (allocation, potential utilization)
- Active frontend workloads (utilization)
- Active web service workloads (utilization)

Resource Assignment and Allocation

- Platform-wide CPU assignment (potential CPU utilization)
- Platform-wide memory assignment (potential memory utilization)
- Platform-wide CPU allocation (active CPU allocation)
- Platform-wide memory allocation (active memory allocation)
- Platform-wide CPU utilization (utilization)
- Platform-wide memory utilization (utilization)
- Platform-wide assignment/allocation differential
- Platform-wide assignment/utilization differential

Error Logging

- Total Errors
- Errors by application version
- Errors by server
- Errors by code source
- Errors by Developer group
- Errors by end user (if known)

Scope: Developer Group

Metrics visible to Developer groups assist Developers in making resource policy assignment decisions, monitoring actual resource utilization, and troubleshooting application errors. For example, by monitoring CPU utilization of a specific application component a Developer group may identify resource bottlenecks in their application, leading to the assignment of a larger resource allocation policy, or code optimization resulting in smaller resource requirements.

Published application versions

- Published web app components
- Published web service components
- Published web app workloads
- Published web service workloads

Sandboxed application versions

- Sandboxed web app components
- Sandboxed web service components
- Sandboxed web app workloads
- Sandboxed web service workloads

Error Logging

- Total Errors
- Errors by application version
- Errors by code source
- Errors by end user (if known)

Resource Assignment and Allocation

- Total resource assignment
- Resource assignment to Published app components
- Resource assignment per application version
- Total resource allocation
- Resource allocation to Published app workloads
- Resource allocation per application version
- Resource assignment / allocation differential

Resource Utilization

- Average CPU utilization across all apps
- Average memory utilization across all apps
- Average CPU utilization per frontend workload
- Average CPU utilization per web service workload
- Average memory utilization per frontend workload
- Average memory utilization per web service workload
- Resource allocation / utilization differential

The Apprenda Platform Allocation Reporting API

When the platform has workload throttling enabled, a web service called the Allocation Reporting API will provide data about the various workloads being distributed across the platform's servers. This web service can be reached using SOAP or REST. The real-time information is extremely useful to operators of the platform in scenarios where

they can action on this allocation information. For example, using the data available from this API, Platform Administrators can discover when individual application components are deployed as workloads on servers, how long they run, and when they shutdown. Using data from attached resource policies, this information tells operators precisely how much of their entire platform infrastructure is allocated for use by guest applications at any queried time.

With the information available through this API, a Platform Administrator can integrate with existing reporting systems or construct helpful external applications, such as the simplistic reporting example here:

RESOURCE ALLOCATION API SAMPLE

Search Criteria

Apprenda Platform Base URL:

Start Time (UTC):* End Time (UTC):

Developer Alias: Application Alias:

NOTE: If both a developer alias and an application alias is supplied only the developer alias will be used in the search.

Show ID Columns:

Application ID	Application Name	Application Alias	Developer ID	Developer Name	Developer Alias	Version ID	Version Name	Version Alias	Component Name	Component Type	Deploy Time (UTC)	Undeploy Time (UTC)	Resource Allocation Policy Name	Memory Limit (MB)	CPU Limit (MHz)	CPU Cores
ffe5f1cb-3f87-498c-92a8-9f8000deabbd	No Application Services PODNA	noappservicespodna	e5dac877-f99a-4eea-8c6d-62884de31b35	PODNA Apps	podna	2a79f273-5514-4f1c-bf32-9f8000e0e14	Version 3	v3	Database	Database	10/18/2011 5:36:45 PM					
ffe5f1cb-3f87-498c-92a8-9f8000deabbd	No Application Services PODNA	noappservicespodna	e5dac877-f99a-4eea-8c6d-62884de31b35	PODNA Apps	podna	2a79f273-5514-4f1c-bf32-9f8000e0e14	Version 3	v3	Provider Database	Database	10/18/2011 5:36:46 PM					
ffe5f1cb-3f87-498c-92a8-9f8000deabbd	No Application Services PODNA	noappservicespodna	e5dac877-f99a-4eea-8c6d-62884de31b35	PODNA Apps	podna	3cfee59d-2ed5-4d40-99ab-9f8000df696e	Version 2	v2	Database	Database	10/18/2011 5:34:59 PM					
ffe5f1cb-3f87-498c-92a8-9f8000deabbd	No Application Services PODNA	noappservicespodna	e5dac877-f99a-4eea-8c6d-62884de31b35	PODNA Apps	podna	3cfee59d-2ed5-4d40-99ab-9f8000df696e	Version 2	v2	Provider Database	Database	10/18/2011 5:34:59 PM					
a4c1c548-da2a-40f1-936f-9f8000e15f11	Authorization PODNA	authorizationpodna	e5dac877-f99a-4eea-8c6d-62884de31b35	PODNA Apps	podna	cb9d9b36-1413-4d2a-844d-9f8000e15f11	Version 1	v1	Provider Database	Database	10/18/2011 5:43:33 PM					
1da86436-231f-47ca-956b-9f8000e0711f	Authentication PODNA	authenticationpodna	e5dac877-f99a-4eea-8c6d-62884de31b35	PODNA Apps	podna	1e6793f0-fe31-4e6a-84c2-9f8000e0eeec	Version 2	v2	Service.Service	Service	10/18/2011 7:13:33 PM	10/18/2011 7:17:30 PM	New Default	500	500	1
1da86436-231f-47ca-956b-9f8000e0711f	Authentication PODNA	authenticationpodna	e5dac877-f99a-4eea-8c6d-62884de31b35	PODNA Apps	podna	1e6793f0-fe31-4e6a-84c2-9f8000e0eeec	Version 2	v2	User Interface	UserInterface	10/18/2011 5:40:08 PM		New Default	500	500	1

Notice the information captured in this report:

- Application
- Application Version
- Developer
- Application Component (frontend or service)
- Deployment time
- Undeployment time
- Attached Resource Allocation Policy
 - Memory
 - CPU
 - CPU Cores

Additionally, if chargeback costs are associated with resource policies, this information can be used to calculate running time for specific application components, which is helpful in calculating chargeback amounts for Developer groups.

You can find an example of this API (the one described above) in our [support website](#).

Source URL: <http://docs.apprenda.com/3-0/administrators>