**Feature Stapling in the SharePoint add-in model**

The approach you take to run code and deploy artifacts when a SharePoint site is provisioned is different in the new SharePoint Add-in model than it was with Full Trust Code. In a typical Full Trust Code (FTC) / Farm Solution scenario, out-of-the-box site definitions were modified with stapled features. Features were used to package and deploy artifacts, configurations, and branding assets associated with a SharePoint site and features were stapled to the site definition. Then the stapled features were automatically installed and activated upon site provisioning.+

In an SharePoint Add-in model scenario, you may staple features, staple Add-ins, or use the SharePoint Client Side Object Model (CSOM) to create and configure site collections and sub sites then deploy artifacts, configurations, and branding assets to them. This pattern is commonly referred to as the remote provisioning pattern.

High Level Guidelines

As a rule of a thumb, we would like to provide the following high level guidelines to create and configure site collections and sub sites then deploy artifacts, configurations, and branding assets to them.

* The only way you can still use feature stapling is when you are stapling features to site collections and you are using sandbox solutions to deploy the site definitions and the stapled features.
* You can use the Add-in stapling model with tenant deployed Add-ins to implement functionality similar to feature stapling.
* You can use the remote provisioning pattern to implement functionality similar to feature stapling by activating additional features on top of the out-of-the-box box site definition via remote APIs.

Options to create and configure site collections and sub sites then deploy artifacts, configurations, and branding assets to them

You have a few options to to create and configure site collections and sub sites then deploy artifacts, configurations, and branding assets to them.

* Staple features
* Staple Add-ins
* Use the remote provisioning pattern

Staple features

In this pattern you staple features to site definitions.

* This pattern is only available at the site collection level.
* It is not possible to staple features to sub sites.
* This is not an optimal or recommended approach because it uses deprecated sandbox solutions and does not set you up well for upgrades.

When is it a good fit?

When you are migrating legacy code in an on-premises SharePoint environment and you do not have time to re-write it properly.

## Staple Add-ins

In this pattern you deploy Add-ins stored in the App Catalog to specific site collections, managed paths, and site templates.

* See the [SharePoint 2013 App Deployment through 'App Stapling' (MSDN Blog Article - Richard DiZerega)](http://blogs.msdn.com/b/richard_dizeregas_blog/archive/2013/09/18/10399333.aspx) for more details about the Add-in stapling model.
* Because the add-in is pushed by an administrator, site owners will not be able to remove the add-in from a site that meets the deployment criteria. Not even a site collection administrator can remove the add-in.
* This centralized deployment also shares the same centralized add-in resources (Add-in Web and Remote Web). Essentially, the Add-in is deployed, but not installed in the sites. All sites will leverage the Add-in Web and Remote Web from the instance installed in the App Catalog.
* Because of centralized deployment, remote events such as 'Handle App Installed', 'Handle App Uninstalled', and 'Handle App Upgrade' will only fire once (when the Add-In is installed in the App Catalog).
  + This can make it difficult to use the Add-in stapling pattern to automatically apply changes to sites where it is deployed because these events do not fire when it is deployed to sites.
* Add-in parts are not supported when Add-ins are stapled to sites.
* This pattern requires manual user actions to deploy the Add-ins.

## Use the remote provisioning pattern

In this pattern you use the SharePoint Client-Side Object Model (CSOM) to create and configure site collections and sub sites then deploy artifacts, configurations, and branding assets to them.

* This pattern does not require packaging artifacts, configurations, and branding assets in separate features, or Add-ins. Everything may be packaged in a single Add-in.
* When you use this pattern for site provisioning you typically override the out of the box page to create a new site.
* For more information about this pattern see the [Site Provisioning (SharePoint Add-in Recipe)](https://docs.microsoft.com/en-us/sharepoint/dev/solution-guidance/site-provisioning-sharepoint-add-in)
* If you wish to deploy Add-ins to a SharePoint site, this can be done via CSOM. Here is an example which loads an Office Add-in via a .app manifest file and installs it in a SharePoint site.