**Introducción**

**Escenario**

Su empresa debe asegurarse de que las implementaciones de máquinas virtuales sean coherentes en toda la organización.

Las plantillas de Resource Manager se usan para implementar recursos, incluidas las máquinas virtuales.

**Aptitudes evaluadas**

La implementación de recursos mediante plantillas de Resource Manager forma parte del [Examen AZ-104: Administrador de Microsoft Azure](https://learn.microsoft.com/es-es/certifications/exams/az-104).

Implementación y administración de recursos de procesos de Azure (20-25 %)

Automatización de la implementación de máquinas virtuales (VM) mediante plantillas de Azure Resource Manager

* Modificar una plantilla de Resource Manager.
* Implementar a partir de una plantilla.
* Guardar una implementación como una plantilla de Resource Manager.

**Objetivos de aprendizaje**

En este módulo aprenderá a:

* Mostrar una lista de las ventajas de las plantillas de Azure.
* Identificar los componentes del esquema de plantillas de Azure.
* Especificar los parámetros de las plantillas de Azure.
* Buscar y usar plantillas de inicio rápido de Azure.

**Revisión de las ventajas de las plantillas de Resource Manager**

Una **plantilla de Resource Manager** define con exactitud todos los recursos de Resource Manager de una implementación. Una plantilla de Resource Manager se puede implementar en un grupo de recursos con una sola operación.

Usar plantillas de Resource Manager le permitirá realizar implementaciones más rápidas y repetibles. Por ejemplo, ya no tiene que crear una máquina virtual en el portal, esperar a que finalice y, después, crear la siguiente máquina virtual. La plantilla de Resource Manager se encarga automáticamente de toda la implementación.

**Ventajas de la plantilla**

* **Las plantillas mejoran la coherencia**. Las plantillas de Resource Manager proporcionan un lenguaje común para que todos describan sus implementaciones. Independientemente de la herramienta o el SDK que use para implementarla, la estructura, el formato y las expresiones dentro de la plantilla siguen siendo las mismas.
* **Las plantillas permiten expresar implementaciones complejas**. Las plantillas permiten implementar varios recursos en el orden correcto. Por ejemplo, no es conveniente implementar una máquina virtual antes de crear una interfaz de red o un disco de sistema operativo (SO). Resource Manager asigna cada recurso y sus recursos dependientes y crea los recursos dependientes en primer lugar. La asignación de dependencias ayuda a garantizar que la implementación se lleva a cabo en el orden correcto.
* **Las plantillas reducen las tareas manuales y propensas a errores**. Crear y conectar recursos manualmente puede llevar mucho tiempo y es fácil cometer errores. Resource Manager garantiza que la implementación se lleva a cabo de la misma manera cada vez.
* **Las plantillas son código**. Las plantillas expresan sus requisitos mediante código. Considere una plantilla como un tipo de infraestructura como código que se puede compartir, probar y versionar como cualquier otro producto de software. Además, dado que las plantillas son código, se puede crear un "rastro" que permita seguirlas. El código de plantilla documenta la implementación. La mayoría de los usuarios somete a sus plantillas a algún tipo de control de revisión, como GIT. Cuando la plantilla cambia, el historial de revisiones también deja constancia de cómo dicha plantilla (y su implementación) ha evolucionado con el tiempo.
* **Las plantillas fomentan la reutilización**. La plantilla puede contener parámetros que se rellenan cuando la plantilla se ejecuta. Un parámetro puede definir un nombre de usuario, una contraseña, un nombre de dominio, etc. Los parámetros de plantilla permiten crear varias versiones de la infraestructura, como ensayo y producción, mientras se sigue usando exactamente la misma plantilla.
* **Las plantillas se pueden vincular**. Puede vincular plantillas de Resource Manager para que estas se puedan modular. Puede escribir plantillas pequeñas que definan una parte de la solución y, después, combinarlas para crear un sistema completo.
* **Las plantillas simplifican la orquestación**. Para implementar todos los recursos tan solo se necesita implementar la plantilla. Normalmente, esto conllevaría varias operaciones.

**Exploración del esquema de plantillas de Resource Manager**

Las plantillas de Resource Manager se escriben en JSON, lo que permite expresar los datos almacenados como un objeto (como una máquina virtual) en texto. Un documento JSON es, básicamente, una colección de pares clave-valor. Cada clave es una cadena, cuyo valor puede ser lo siguiente:

* Una cadena
* Un número
* Una expresión booleana.
* Una lista de valores.
* Un objeto (que es una colección de otros pares clave-valor).

Una plantilla de Resource Manager puede contener secciones que se expresen mediante la notación JSON, pero no están relacionadas con el propio lenguaje JSON:

{

"$schema": "http://schema.management.​azure.com/schemas/2019-04-01/deploymentTemplate.json#",​

"contentVersion": "",​

"parameters": {},​

"variables": {},​

"functions": [],​

"resources": [],​

"outputs": {}​

}

| **Nombre del elemento** | **Obligatorio** | **Descripción** |
| --- | --- | --- |
| $schema | Sí | Ubicación del archivo de esquema JSON que describe la versión del idioma de la plantilla. Use la dirección URL que se muestra en el ejemplo anterior. |
| contentVersion | Sí | Versión de la plantilla (por ejemplo, 1.0.0.0). Puede especificar cualquier valor para este elemento. Use este valor para documentar los cambios importantes de la plantilla. Este valor se puede usar para asegurarse de que se está usando la plantilla correcta. |
| parámetros | No | Valores que se proporcionan cuando se ejecuta la implementación para personalizar la implementación de recursos. |
| variables | No | Valores que se usan como fragmentos JSON en la plantilla para simplificar expresiones de idioma de la plantilla. |
| functions | No | Funciones definidas por el usuario que están disponibles dentro de la plantilla. |
| resources | Sí | Tipos de servicios que se implementan o actualizan en un grupo de recursos. |
| outputs | No | Valores que se devuelven después de la implementación. |

**Exploración de los parámetros de plantillas de Azure Resource Manager**

En la sección de parámetros de la plantilla, especifique los valores que el usuario puede introducir al implementar los recursos. Las propiedades disponibles para un parámetro son:

"parameters": {

"<parameter-name>" : {

"type" : "<type-of-parameter-value>",

"defaultValue": "<default-value-of-parameter>",

"allowedValues": [ "<array-of-allowed-values>" ],

"minValue": <minimum-value-for-int>,

"maxValue": <maximum-value-for-int>,

"minLength": <minimum-length-for-string-or-array>,

"maxLength": <maximum-length-for-string-or-array-parameters>,

"metadata": {

"description": "<description-of-the parameter>"

}

}

}

Este es un ejemplo que ilustra dos parámetros, uno relacionado con un nombre de usuario de máquina virtual y el otro con su contraseña correspondiente:

"parameters": {

"adminUsername": {

"type": "string",

"metadata": {

"description": "Username for the Virtual Machine."

}

},

"adminPassword": {

"type": "securestring",

"metadata": {

"description": "Password for the Virtual Machine."

}

}

}

**Nota**

Está limitado a 256 parámetros en una plantilla. Puede reducir el número de parámetros mediante el uso de objetos que contienen varias propiedades.

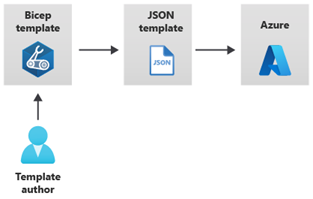
**Consideración de las plantillas de Bicep**

[Azure Bicep](https://learn.microsoft.com/es-es/azure/azure-resource-manager/bicep/overview) es un lenguaje específico de dominio (DSL) que usa una sintaxis declarativa para implementar recursos de Azure. Brinda sintaxis concisa, seguridad de tipos confiable y compatibilidad con la reutilización de código.

Puede usar Bicep en lugar de JSON para desarrollar las plantillas de Azure Resource Manager (plantillas de ARM). La sintaxis JSON para crear una plantilla de ARM puede ser detallada y exigir expresiones complicadas. La sintaxis de Bicep reduce esa complejidad y mejora la experiencia de desarrollo. Bicep es una abstracción transparente del código JSON de plantillas de ARM, y no pierde ninguna de las funcionalidades de las plantillas JSON.

**¿Cómo funciona Bicep?**

Al implementar un recurso o una serie de recursos en Azure, las herramientas integradas en Bicep convierten la plantilla de Bicep en una plantilla JSON. Este proceso se conoce como transpilación. La transpilación es el proceso de convertir el código fuente escrito de un lenguaje a otro.



Bicep proporciona muchas mejoras sobre JSON para la creación de plantillas, entre las que se incluyen las siguientes:

* **Sintaxis más sencilla**: Bicep proporciona una sintaxis más sencilla para escribir plantillas. Puede hacer referencia a parámetros y variables directamente, sin usar funciones complicadas. La interpolación de cadenas se usa en lugar de la concatenación para combinar valores para nombres y otros elementos. Puede hacer referencia a las propiedades de un recurso directamente mediante su nombre simbólico en lugar de instrucciones de referencia complejas. Estas mejoras de sintaxis ayudan tanto con la creación como con la lectura de plantillas de Bicep.
* **Módulos**: puede desglosar las implementaciones de plantillas complejas en archivos de módulo más pequeños y hacer referencia a ellos en una plantilla principal. Estos módulos proporcionan una administración más sencilla y una mayor capacidad de uso.
* **Administración automática de dependencias**: en la mayoría de las situaciones, Bicep detecta automáticamente las dependencias entre los recursos. Este proceso quita parte del trabajo que implica la creación de plantillas.
* **Validación de tipos e IntelliSense**: la extensión Bicep para Visual Studio Code incluye una validación enriquecida e IntelliSense para todas las definiciones de API de tipos de recursos de Azure. Esta característica ayuda a proporcionar una experiencia de creación más sencilla.

**Revisión de plantillas de inicio rápido**

Las [plantillas de inicio rápido de Azure](https://azure.microsoft.com/resources/templates/) son plantillas de Azure Resource Manager que proporciona la comunidad de Azure.



Algunas plantillas proporcionan todo lo necesario para implementar la solución, mientras que otras pueden servir como punto de partida para la plantilla. Sea cual sea su caso, puede estudiar estas plantillas para saber cuál es la mejor forma de crear y estructurar sus propias plantillas.

* El archivo README.md proporciona información general de lo que hace la plantilla.
* El archivo azuredeploy.json define los recursos que se implementarán.
* El archivo azuredeploy.parameters.json proporciona los valores que necesita la plantilla.

**Nota**

Dedique unos minutos a examinar las plantillas disponibles. ¿Le resulta algo interesante?

**Prueba de conocimientos**

Elija la respuesta más adecuada para cada pregunta. Después, seleccione Comprobar las respuestas.

Principio del formulario

**1. ¿Qué es una plantilla de Azure Resource Manager?**

1. Una serie de comandos de la CLI de Azure para implementar la infraestructura en Azure.
2. Un archivo de notación de objetos JavaScript (JSON) que define la infraestructura y la configuración de la implementación.
3. Un script que usa Azure Resource Manager para administrar la cuenta de almacenamiento de Azure.

**2. ¿Cuál de los parámetros siguientes es un elemento del esquema de plantilla?**

1. Includes
2. Scripts
3. Salidas

**3. ¿Qué ocurre si la misma plantilla se ejecuta una segunda vez?**

1. Azure Resource Manager implementa los recursos nuevos como copias de los que ya se hayan implementado.
2. Azure Resource Manager no cambia los recursos implementados.
3. Azure Resource Manager elimina los recursos que ya se hayan implementado y los vuelve a implementar.
4. B
5. C
6. B

Final del formulario

**Resumen y recursos**

Para implementar la infraestructura como código para las soluciones de Azure, use las plantillas de Azure Resource Manager. La plantilla es un archivo de notación de objetos JavaScript (JSON) que contiene la infraestructura y la configuración del proyecto. La plantilla usa sintaxis declarativa, lo que permite establecer lo que pretende implementar sin tener que escribir la secuencia de comandos de programación para crearla. En la plantilla se especifican los recursos que se van a implementar y las propiedades de esos recursos.

Ahora debería poder realizar las siguientes acciones:

* Mostrar una lista de las ventajas de las plantillas de Azure.
* Identificar los componentes del esquema de plantillas de Azure.
* Especificar los parámetros de las plantillas de Azure.
* Buscar y usar plantillas de inicio rápido de Azure.