IBM Cúram Social Program Management

Cúram Cache

Versión 6.0.5

IBM Cúram Social Program Management

Cúram Cache

Versión 6.0.5

Nota

Antes de utilizar esta información y el producto al que hace referencia, lea la información que figura en el apartado "Avisos" en la página 15

Revisión: mayo de 2013

Esta edición se aplica a IBM Cúram Social Program Management v6.0 5 y a todos los releases posteriores hasta que se indique lo contrario en nuevas ediciones.

Contenido

Figuras

Tablas

Capítulo 1. Introducción

1.1 Finalidad

La finalidad de este documento es introducir Cúram Cache, un servicio de almacenamiento en memoria caché genérico diseñado para cumplir el requisito para memorias caché de corta y larga duración en la aplicación.

1.2 Público al que va dirigida

Esta guía está pensada para arquitectos y desarrolladores interesados en la utilización de Cúram Cache para cumplir sus requisitos de almacenamiento en memoria caché.

Capítulo 2. Cúram Cache

2.1 ¿Qué es Cúram Cache?

Cúram Cache es un servicio de almacenamiento en memoria caché genérico diseñado para cumplir el requisito para memorias caché de corta y larga duración en la aplicación. El servicio está disponible en los contenedores de cliente y servidor de un entorno de servidor de aplicaciones (aplicación en línea), así como en un proceso Java™ autónomo estándar (programas por lotes).

Cúram Cache permite la creación de tres tipos de memorias caché:

- Global - son memorias caché globales (en el nivel de JVM).
- Local de hebra - son memorias caché que duran tanto como la hebra a la que pertenecen.
- Local de transacción - son memorias caché que existen mientras dura la transacción actual.

En este documento se hace referencia colectivamente a los dos últimos tipos de memorias caché como memorias caché de varias instancias ya que en cualquier momento puede haber más de una instancia de una memoria caché con un nombre determinado (uno para cada transacción o hebra activa).

2.2 Configuración

La configuración de todos los tipos de memoria caché de Cúram Cache es totalmente declarativa y se basa en el mecanismo de configuración proporcionado por la aplicación. Los parámetros de configuración de memoria caché deben añadirse a la sección de la propiedad `APP_CACHE`.

En la implementación actual, las memorias caché globales dan soporte a políticas de desalojo basadas en el tamaño y el tiempo mientras que las memorias caché de varias instancias sólo dan soporte a una política de desalojo basada en el tiempo.

Se pueden ajustar los siguientes parámetros de configuración de memoria caché:

- Tamaño – el número máximo de elementos en la memoria. El valor predeterminado es 200. El tipo es `INT32`.
- Política de desalojo – la política utilizada para desalojar elementos de la memoria cuando se alcanza el número máximo de elementos en la memoria. El valor predeterminado es `LRU` . El tipo es `STRING` . Los valores válidos son:
  - `LRU` – utilizado menos recientemente
  - `LFU` – utilizado menos frecuentemente
  - `FIFO` – primero en entrar, primero en salir
- Tiempo de vida - el número de segundos que un elemento puede vivir en una memoria caché. Cuando se establece en un valor positivo distinto de cero, se descartan todos los elementos que han estado en la memoria caché un tiempo superior al valor de este parámetro, en segundos.
- Tiempo de desocupado - el número de segundos que un elemento de la memoria caché puede estar sin utilizar antes de ser descartado. Cuando se establece en un valor positivo distinto de cero, se descartan todos los elementos sin utilizar durante más del valor de este parámetro, en segundos.

Todas las propiedades de configuración de memoria caché deben ajustarse a esta notación:

```
curam.cache.<nombre_grupo_memoriascaché>.<nombre_memoriacaché>.<parámetro>
```

donde:

- <nombre_grupo_memoriascaché> - es el nombre del grupo de memorias caché al que pertenece la memoria caché.
- <nombre_memoriacaché> - es el nombre de la memoria caché. También podría ser "*" y entonces, el parámetro de configuración se aplica a todas las memorias caché de este grupo de memorias caché.
- <parámetro> - puede ser size , evictionPolicy, timeToIdle o timeToLive.

En el ejemplo siguiente, la memoria caché global curam.myproject.mycache del grupo de memorias caché globales curam-group se ha configurado con un tamaño de 1000 elementos y una política de desalojo de Utilizado menos recientemente.

```
curam.cache.curam-group.curam.myproject.mycache.size=1000
curam.cache.curam-group.curam.myproject.mycache.evictionPolicy=LRU
```

Figura 1. Configuración de una memoria caché

En este segundo ejemplo, la memoria caché local de transacciones curam.myproject.mycache del grupo de memorias caché locales de transacciones transaction-group se ha configurado con un tiempo de desocupado de 10 segundos mientras que las demás memorias caché locales de transacciones se han configurado con un valor de 5 segundos.

```
curam.cache.transaction-group.curam.myproject.mycache
                                        .timeToIdle=10
curam.cache.transaction-group.*.timeToIdle=5
```

Figura 2. Configuración de todas las memorias caché en un grupo

Los datos de configuración de memoria caché almacenados en el repositorio de configuración de aplicación se pueden anular pasando los valores relevantes como propiedades del sistema JVM. Esto puede ser interesante para los procesos por lotes en que el perfil de aplicación puede ser distinto de la aplicación en línea.

El ejemplo siguiente muestra cómo inhabilitar la memoria caché global curam.myproject.mycache del grupo de memorias caché globales para un proceso por lotes.

```
ant -f app_batchlauncher.xml
     -Dcuram.cache.curam-group.curam.myproject.mycache.size=0
     -Dbatch.userna...
```

Figura 3. Inhabilitación de una memoria caché para un proceso por lotes

2.3 Estadísticas

Todas las memorias caché de Cúram Cache se han instrumentado para estadísticas y éstas se integran con la infraestructura de Cúram JMX. El siguiente conjunto mínimo de estadísticas se expone para cada tipo de memoria caché mediante el MBean CuramCacheStats:

- Grupo de memorias caché - el nombre de cada grupo de memorias caché
- Memoria caché – el nombre de la memoria caché
- Capa – el nombre de la capa de memoria caché (memoria, disco,...)
- Tamaño – el número de elementos en la memoria caché
- Coincidencias - el número de solicitudes a la memoria caché que han devuelto un elemento ya cargado en la memoria caché
- Fallos - el número de solicitudes a la memoria caché que han devuelto un elemento que tenía que cargarse en la memoria caché
- Desalojos - el número de veces que los elementos se han desalojado de la memoria caché

- Tiempo promedio de obtención (ns) - el tiempo promedio transcurrido, en nanosegundos, que se tarda en leer un elemento de la memoria caché. Tenga en cuenta que algunos proveedores de memoria caché sólo dan soporte a una resolución de milisegundos.

Las memorias caché de varias instancias ofrecen estadísticas instantáneas y agregadas. Las estadísticas instantáneas son para todas las instancias activas en el momento de la consulta y las estadísticas agregadas se calculan a partir de todas las instancias que se han creado.

2.4 Cierre de Cúram Cache

Cúram Cache requiere un cierre ordenado al salir de JVM. Cúram Cache instala automáticamente un gancho de cierre de JVM para borrar la memoria caché como una solución de último recurso pero, siempre que sea posible, se recomienda utilizar el cierre explícito invocando `CacheManager.shutdown()` al cerrar la aplicación.

Capítulo 3. Memorias caché globales

3.1 Introducción

Las memorias caché globales son memorias caché que existen en el ámbito del proceso de JVM o fuera de éste. En la versión actual de Cúram Cache, las memorias caché globales existen únicamente en el ámbito del proceso de JVM. Una entrada almacenada en una memoria caché global dura a través de los límites de la transacción hasta que es eliminada explícitamente por el desarrollador o se elimina implícitamente como resultado de la política de desalojo asociada con la memoria caché.

Es importante tener en cuenta que dado que las memorias caché globales son de larga duración, sus datos son propensos a presentar periodos cortos de incoherencia cuando se actualizan los objetos almacenados en memoria caché. Cuando se efectúa una actualización en la aplicación que afecta a un objeto almacenado en memoria caché, la entrada de memoria caché asociada se invalida de manera asíncrona. La infraestructura de almacenamiento en memoria caché garantiza que la entrada de memoria caché se invalida finalmente, pero no puede garantizar un margen de tiempo máximo determinado. La comprensión de este comportamiento es muy importante al decidir si determinados datos de la aplicación se pueden almacenar en una memoria caché global.

3.2 Proveedor de memoria caché global

Cúram Cache implementa gran parte de la infraestructura de almacenamiento en memoria caché global utilizando soluciones de almacenamiento en memoria caché de terceros, a las cuales se hace referencia en este documento como proveedores de almacenamiento en memoria caché. El proveedor predeterminado es Ehcache, una infraestructura de almacenamiento en memoria caché distribuida, de alto rendimiento y de código abierto.

3.3 Grupo de memorias caché globales predeterminado

Las memorias caché globales se agrupan basándose en los requisitos de configuración comunes como, por ejemplo, de duplicación y almacenamiento en disco. Todas las memorias caché de la aplicación deberían crearse en el grupo de memorias caché predeterminado. El nombre del grupo de memorias caché predeterminado es `curam-group`.

En la implementación actual, el grupo de memoria caché predeterminado no es de duplicación automática y no da soporte a desbordamiento de disco y persistencia de disco. Puesto que la duplicación automática está inhabilitada, las operaciones de memoria caché sólo son visible para la JVM en que se encuentra la memoria caché global. Sin embargo, para mantener de forma coherente todas las memorias caché del grupo de memorias caché predeterminado en todo el clúster del servidor de aplicaciones, se proporciona un mecanismo de invalidación de memoria caché explícito. La invalidación de memoria caché sólo la puede desencadenar el código de servidor, pero la invalidación de memoria caché invalida las memorias caché de los contenedores de servidor y de cliente en todas las JVM del clúster del servidor de aplicaciones.

3.4 Memorias caché globales

Las memorias caché globales se crean con una llamada al método get() del grupo de memorias caché. Si una memoria caché todavía no existe, se crea una memoria caché y, si existen datos de configuración, se aplican a ésta antes de que se devuelva. Las memorias caché globales normalmente se llenan utilizando cargadores de memoria caché registrados por los clientes de memoria caché. Este método aisla el cliente de memoria caché de la gestión del acceso simultáneo a la memoria caché mientras de carga la memoria caché.

Cúram Cache no impone la utilización de objetos serializables en su API, sin embargo determinadas características ofrecidas por la infraestructura de almacenamiento en memoria caché sólo están disponibles si la clave o el objeto almacenado en memoria caché son serializables. Por este motivo, se recomienda, siempre que sea posible, utilizar claves y valores serializables en Cúram Cache.

Utilización de claves no serializables: Las entradas de memoria caché que tienen claves no serializables sólo se invalidan en la JVM local y no en todo el clúster del servidor de aplicaciones.

3.5 Recomendaciones de utilización

A continuación se proporciona una lista de recomendaciones sobre cómo debe utilizarse una memoria caché global:

- Sólo almacene en memoria caché objetos inmutables.
- Utilice claves y valores serializables siempre que sea posible. Como mínimo, las claves deben ser serializables.
- Utilice un cargador de memoria caché para llenar la memoria caché. Esto permite que la memoria caché se beneficie de las optimizaciones en la simultaneidad de grano fino incorporadas en el proveedor de memoria caché y no es necesario que el usuario se preocupe de gestionar el acceso simultáneo a la memoria caché.
- La carga de una memoria caché sin un cargador (utilizando las llamadas de `get()` y `put()`) debe evitarse por dos motivos principales:
  - Gestión de la simultaneidad - En este caso, el usuario es responsable de gestionar el acceso simultáneo a la memoria caché mientras de carga la memoria caché. El usuario tiene dos opciones:
    - Controlar el acceso simultáneo al bloque de código de `get()` y `put()` – este método no es recomendable en una parte sensible al rendimiento de la aplicación pero proporciona la garantía de que un objeto sólo se carga una vez.
    - Permitir el acceso simultáneo al bloque de código de `get()` y `put()` – este método da soporte a una mayor simultaneidad pero un objeto se puede cargar más de una vez por parte de diferentes hebras.
  - Gestión de datos eficaz - sin un cargador de memoria caché, la memoria caché debe llenarse previamente con todos los datos. Con un cargador de memoria caché, sólo se llena la memoria caché con los datos necesarios.
- Utilice nombres de memoria caché que tengan como prefijo un nombre de paquete exclusivo de su proyecto. Por ejemplo, `curam.cpm.myCache` sería un nombre adecuado para una memoria caché en el proyecto de Curam Provider Management™.

3.6 Configuración

Todas las memorias caché globales heredan los siguientes valores predeterminados para los parámetros de configuración.

- `size` - 200
- `evictionPolicy` - LRU
- `timeToLive` - 0 (no está activo)
- `timeToIdle` - 0 (no está activo)

Estos valores se pueden anular para cualquier memoria caché global. En el ejemplo siguiente `curam-group` es el nombre del grupo de memorias caché predeterminado y `curam.myproject.mycache` es el nombre de la memoria caché.

```
curam.cache.curam-group.curam.myproject.mycache.size=1000
curam.cache.curam-group.curam.myproject.mycache.evictionPolicy=LRU
curam.cache.curam-group.curam.myproject.mycache.timeToLive=3600
curam.cache.curam-group.curam.myproject.mycache.timeToIdle=300
```

Figura 4. Configuración de una memoria caché global

3.7 Utilización de memorias caché globales en un contexto transaccional

Cuando se utiliza una memoria caché global en un contexto transaccional, debe tenerse cuidado para asegurarse de que la memoria caché mantiene su coherencia en caso de que se retrotraiga la transacción actual.

Invalidación de memoria caché en un contexto transaccional: Cuando se modifican datos que afectan al contenido de una memoria caché, no debe eliminarse o actualizarse directamente el elemento almacenado en memoria caché; en lugar de ello, debe invocarse el método `CacheManagerEjb.postInvalidationMessage()` para enviar un mensaje de invalidación que desencadenará la invalidación de memoria caché.

3.8 Ejemplos de código

Esta sección contiene ejemplos de código que muestran cómo escribir un cargador de memoria caché, cómo utilizar la memoria caché y cómo invalidar una entrada de memoria caché.

3.9 Cargador de memoria caché

En el ejemplo siguiente, se utiliza la clase CacheLoaderAdapter para ayudar en la implementación de MyCacheLoader.

```
...
public class MyCacheLoader extends
    CacheLoaderAdapter<Integer, ReadWorkQueueDetails> {
  /* (no Javadoc)
   * @ver curam.util.cache.CacheLoader#load(java.lang.Object)
   */
  public ReadWorkQueueDetails load(Integer workQueueID)
      throws AppException, InformationalException {
    WorkAllocation wa = (WorkAllocation)WorkAllocationFactory
                                .newInstance();
    ReadWorkQueueKey key = new ReadWorkQueueKey();
    key.key = new ReadWorkQueueKey();
    key.key.key = new WorkQueueKey();
    key.key.key.workQueueID = workQueueID;
    ReadWorkQueueDetails item = wa.readWorkQueue(key);
    if(item != null) {
      return item.dtls;
    }
    return null;
  }
}
...
```

Figura 5. Utilización de CacheLoaderAdapter para implementar un cargador de memoria caché

3.10 Cliente de memoria caché

El ejemplo siguiente muestra el modo habitual de registrar un cargador de memoria caché y de utilizar la memoria caché.

```
...
public class MyCacheClient {
  // mantener una referencia estática de mycache
  private static Cache<Integer,
            ReadWorkQueueDetails> myCache;

  static {
    // recuperar una referencia de mycache y registrar
    // el cargador de memoria caché
    myCache = CacheManager.getDefaultCacheGroup()
                               .getCache("mycache");
    myCache.registerCacheLoader(new MyCacheLoader());
  }

  public WorkAllocation() {
...
  }
...
  // utiliza la memoria caché
  ReadWorkQueueDetails wq = myCache.get(1);
...
```

Figura 6. Registro de un cargador de memoria caché y utilización de la memoria caché

3.11 Invalidación de memoria caché

El ejemplo siguiente muestra cómo invalidar una entrada de memoria caché en el código que se ejecuta en un contexto transaccional (código de servidor). Como se ha explicado en 3.3, “Grupo de memorias caché globales predeterminado”, en la página 7, la invalidación de memoria caché para memorias caché globales del grupo de memorias caché predeterminado sólo la puede desencadenar el código de servidor.

```
...
CacheManagerEjb.postInvalidationMessage(
          new CacheInvalidationMessage<Key>("mycache", 1));
...
```

Figura 7. Invalidación de una entrada de memoria caché en el código de servidor

Capítulo 4. Memorias caché locales de hebras

4.1 Visión general

Estas memorias caché están estrechamente vinculadas a la hebra que se ha utilizado para crearlas. Ninguna otra hebra puede acceder a los datos de estas memorias caché y las memorias caché sólo se destruyen cuando finaliza la hebra que las ha creado. Las memorias caché locales de hebra están muy especializadas. Sólo se deben utilizar para memorias caché pequeñas en que la sobrecarga del control de acceso de varias hebras que existe para la memoria caché global no se puede soportar.

4.2 Configuración

Las memorias caché locales de hebras sólo dan soporte a una política de desalojo basada en el tiempo. Los dos únicos parámetros de configuración que se pueden utilizar y sus valores predeterminados son:

- `timeToLive` - 0 (no está activo)
- `timeToIdle` - 0 (no está activo)

El nombre del grupo para las memorias caché locales de hebras es `thread-group`. Debe utilizarse este nombre para configurar la memoria caché local de hebras, tal como se muestra en el ejemplo siguiente.

```
curam.cache.thread-group.curam.myproject.mycache.timeToLive=60
curam.cache.thread-group.curam.myproject.mycache.timeToIdle=10
```

Figura 8. Configuración de una memoria caché local de hebras con el nombre `curam.myproject.mycache`

4.3 Ejemplos de código

Las memorias caché locales de hebras sólo deben accederse cuando existe el contexto (hebra) correcto. Por ejemplo, no se recomienda establecer una memoria caché local de hebras en un bloque de código estático ya que la hebra puede no ser la misma que la hebra que utilice la memoria caché posteriormente.

```
public void myMethod() {
  ...
  Cache<String, String> threadCache = CacheManager.
           getThreadLocalCacheGroup().getCache("mycache");
  String value = threadCache.get("key");
  if(value == null) {
    // realizar operación costosa para calcular el valor - este
    // proceso sólo ocurre una vez para cada hebra
    ...
    // y almacenar el resultado
    threadCache.put("key", "value");
  }
  ...
}
```

Figura 9. Configuración y utilización de una memoria caché local de hebras

Capítulo 5. Memorias caché locales de transacciones

5.1 Visión general

Una memoria caché local de transacciones es una memoria caché que existe mientras dura la transacción actual. Este tipo de memoria caché sólo está disponible en la aplicación de servidor.

5.2 Configuración

Las memorias caché locales de transacciones sólo dan soporte a una política de desalojo basada en el tiempo. Los dos únicos parámetros de configuración que se pueden utilizar y sus valores predeterminados son:

- `timeToLive` - 0 (no está activo)
- `timeToIdle` -5

El nombre del grupo para las memorias caché locales de transacciones es `transaction-group`. Debe utilizarse este nombre para configurar las memorias caché locales de hebras, tal como se muestra en el ejemplo siguiente.

```
curam.cache.transaction-group.curam.myproject.mycache
                                          .timeToLive=60
curam.cache.transaction-group.curam.myproject.mycache
                                          .timeToIdle=10
```

Figura 10. Configuración de una memoria caché local de transacciones con el nombre `curam.myproject.mycache`

5.3 Ejemplos de código

Al igual que las memorias caché locales de hebras, las memorias caché locales de transacciones sólo deben accederse cuando existe el contexto (transacción) correcto.

```
public void myMethod() {
  ...
  Cache<String, String> txnCache = CacheManagerEjb.
           getTransactionLocalCacheGroup().getCache("mycache");
  String value = txnCache.get("key");
  if(value == null) {
    // realizar operación costosa para calcular el valor - este
    // proceso sólo ocurre una vez por transacción
    ...
    // y almacenar el resultado
    txnCache.put("key", "value");
  }
  ...
}
```

Figura 11. Configuración y utilización de una memoria caché local de transacciones

Avisos

Esta información se ha desarrollado para productos y servicios ofrecidos en los Estados Unidos. Es posible que IBM no ofrezca los productos, servicios o características que se describen en este documento en otros países. Solicite información al representante local de IBM acerca de los productos y servicios disponibles actualmente en su zona. Cualquier referencia a un producto, programa o servicio de IBM no pretende afirmar ni implica que sólo pueda utilizarse ese producto, programa o servicio de IBM. En su lugar, se puede utilizar cualquier producto, programa o servicio funcionalmente equivalente que no vulnere ningún derecho de propiedad intelectual de IBM. No obstante, es responsabilidad del usuario evaluar y verificar el funcionamiento de cualquier producto, programa o servicio que no sea de IBM. IBM podría tener patentes o solicitudes de patentes pendientes relacionadas con el tema principal que se describe en este documento. La posesión de este documento no confiere ninguna licencia sobre dichas patentes. Puede enviar consultas sobre licencias, por escrito, a:

IBM Director of Licensing

IBM Corporation

North Castle Drive

Armonk, NY 10504-1785

EE.UU.

Para consultas sobre licencias relacionadas con información de doble byte (DBCS), póngase en contacto con el departamento de propiedad intelectual de IBM de su país o envíe sus consultas, por escrito, a:

Intellectual Property Licensing

Legal and Intellectual Property Law.

IBM Japan Ltd.

19-21, Nihonbashi-Hakozakicho, Chuo-ku

Tokio 103-8510, Japón

El párrafo siguiente no se aplica al Reino Unido ni a ningún otro país donde las disposiciones en él expuestas sean incompatibles con la legislación local: INTERNATIONAL BUSINESS MACHINES CORPORATION PROPORCIONA ESTA PUBLICACIÓN "TAL CUAL" SIN GARANTÍA DE NINGUNA CLASE, YA SEA EXPLÍCITA O IMPLÍCITA, INCLUIDAS, PERO SIN LIMITARSE A ELLAS, LAS GARANTÍAS IMPLÍCITAS DE NO VULNERABILIDAD, COMERCIALIZACIÓN O IDONEIDAD PARA UN PROPÓSITO DETERMINADO. Algunos países no permiten la renuncia a garantías explícitas o implícitas en determinadas transacciones, por lo que puede que esta declaración no sea aplicable en su caso.

La información de este documento puede incluir imprecisiones técnicas o errores tipográficos. Periódicamente se efectúan cambios en la información aquí contenida; estos cambios se incorporarán en nuevas ediciones de la publicación. IBM puede reservarse el derecho de realizar mejoras y/o cambios en los productos y/o programas descritos en esta publicación en cualquier momento sin previo aviso.

Cualquier referencia incluida en esta información a sitios web que no sean de IBM sólo se proporciona para su comodidad y en ningún modo constituye una aprobación de dichos sitios web. El material de esos sitios web no forma parte del material de este producto de IBM y la utilización de esos sitios web se realizará bajo su total responsabilidad.

IBM puede utilizar o distribuir cualquier información que se le facilite de la manera que considere adecuada, sin contraer por ello ninguna obligación con el remitente. Los titulares de licencias de este programa que deseen obtener información sobre el mismo con el fin de permitir: (i) el intercambio de información entre programas creados independientemente y otros programas (incluido éste) y el uso mutuo de información que se haya intercambiado, deben ponerse en contacto con:

IBM Corporation

Dept F6, Bldg 1

294 Route 100

Somers NY 10589-3216

EE.UU.

Dicha información puede estar disponible, sujeta a los términos y condiciones apropiados, incluido en algunos casos el pago de una cuota.

IBM proporciona el programa bajo licencia que se describe en este documento y todo el material bajo licencia disponible para el mismo bajo los términos del Acuerdo de cliente de IBM, el Acuerdo internacional de licencias de programas de IBM o cualquier acuerdo equivalente entre las partes.

Los datos de rendimiento incluidos aquí se determinaron en un entorno controlado. Por lo tanto, los resultados obtenidos en otros entornos operativos pueden variar considerablemente. Algunas mediciones podrían haberse realizado en sistemas en desarrollo y, por lo tanto, no existe ningún tipo de garantía de que dichas mediciones sean las mismas en los sistemas con disponibilidad general. Además, es posible que algunas mediciones se hayan calculado mediante extrapolación. Los resultados reales pueden variar. Los usuarios de este documento deben verificar los datos aplicables a sus entornos específicos.

La información relacionada con productos que no son de IBM se ha obtenido de los proveedores de dichos productos, de sus anuncios publicados o de otras fuentes de disponibilidad pública.

IBM no ha probado estos productos y no puede confirmar la precisión de rendimiento, compatibilidad ni otras afirmaciones relacionadas con productos que no son de IBM. Las preguntas relativas a las prestaciones de productos que no son de IBM deben dirigirse a los proveedores de dichos productos.

Las afirmaciones relativas a las intenciones futuras de IBM están sujetas a cambio o retirada sin previo aviso, y sólo representan objetivos.

Todos los precios de IBM que se muestran son precios de distribuidor recomendados por IBM, corresponden al momento actual y están sujetos a cambios sin aviso previo. Los precios de los distribuidores pueden variar.

Esta información se ofrece con fines de planificación únicamente. La información incluida en este documento puede cambiar antes de que los productos descritos estén disponibles.

Esta información contiene ejemplos de datos e informes utilizados en operaciones comerciales diarias. Para ilustrarlos de la manera más completa posible, los ejemplos incluyen los nombres de personas, empresas, marcas y productos. Todos estos nombres son ficticios y cualquier parecido con nombres y direcciones utilizados por empresas comerciales reales son mera coincidencia.

LICENCIA DE COPYRIGHT:

Esta información contiene programas de aplicación de ejemplo en lenguaje fuente, que ilustran técnicas de programación en diversas plataformas operativas. Puede copiar, modificar y distribuir los programas de ejemplo de cualquier forma, sin tener que pagar a IBM, con intención de desarrollar, utilizar, comercializar o distribuir programas de aplicación que estén en conformidad con la interfaz de programación de aplicaciones (API) de la plataforma operativa para la que están escritos los programas de ejemplo. Estos ejemplos no se han probado exhaustivamente bajo todas las condiciones. Por lo tanto, IBM no puede garantizar ni implicar la fiabilidad, capacidad de servicio o función de estos programas. Los programas de ejemplo se proporcionan "TAL CUAL", sin garantía de ningún tipo. IBM no es responsable de ningún daño resultante de la utilización de los programas de ejemplo por parte del usuario.

Todas las copias o fragmentos de las copias de estos programas de ejemplo o cualquier trabajo que de ellos se derive, deberán incluir un aviso de copyright como el que se indica a continuación:

© (el nombre de la empresa) (año). Algunas partes de este código proceden de los programas de ejemplo de IBM Corp.

© Copyright IBM Corp. _escriba el año o los años_. Reservados todos los derechos.

Si visualiza esta información en una copia software, es posible que no aparezcan las fotografías ni las ilustraciones en color.

Información sobre interfaces de programación

Esta publicación documenta las interfaces de programación que permiten al cliente escribir programas para obtener los servicios de IBM Cúram Social Program Management.

Marcas registradas

IBM, el logotipo de IBM e ibm.com son marcas registradas de International Business Machines Corp., registradas en muchas jurisdicciones en todo el mundo. Otros nombres de productos y servicios pueden ser marcas registradas de IBM u otras empresas. Encontrará una lista actual de marcas registradas de IBM en la web en "Copyright and trademark information" en http://www.ibm.com/legal/us/en/copytrade.shtml.

Java y todas las marcas registradas y logotipos basados en Java son marcas registradas de Oracle y/o sus filiales.

Otros nombres pueden ser marcas registradas de sus respectivos propietarios. Otros nombres de empresas, productos o servicios pueden ser marcas registradas o de servicio de terceros.

Impreso en España