

Manual Técnico

Tema: Servicio Emails

Tabla de contenido

[Introducción 3](#_Toc473414110)

[Operaciones del servicio 5](#_Toc473414111)

[sendText 5](#_Toc473414112)

[sendHtml 6](#_Toc473414113)

[findByEmail 8](#_Toc473414114)

[Cliente JAVA 9](#_Toc473414115)

[sendText 9](#_Toc473414116)

[sendHtml 10](#_Toc473414117)

[sendTextSynchronized 11](#_Toc473414118)

[sendHtmlSynchronized 12](#_Toc473414119)

[findByEmail 13](#_Toc473414120)

Introducción

Este servicio fue creado para centralizar el envio de emails en un solo lugar y evitar que se deba impementar en cada caso dicha funcionalidad. El servicio permite enviar correos electrónicos en formato texto o formato html y agregarle a estos archivos adjuntos que no excedan el máximo permitido por el servidor de correos utilizado en el banco.

Los parámetros que utiliza el servicio son parametrizables desde la base de datos BBVA\_BONITA en la tabla WS\_PARAMETER, en la misma se encuentran los parámetros:

* EMAIL\_SERVER\_HOST
* EMAIL\_SERVER\_PORT
* EMAIL\_SERVER\_USERNAME
* EMAIL\_SERVER\_PASSWORD
* EMAIL\_SERVER\_USERNAME\_ALIAS
* EMAIL\_SERVER\_AUTHENTICATION\_MODE

Los valores de estos parámetros indicar al servicio que servidor de correo utilizar, que cuenta y que tipo de autenticación.

El mismo ha sido desarrollado para dar soporte a la nueva plataforma de procesos de bonita 7 y se encuentra desplegado en el AS del Carbon.

Arquitectura

Este servicio está desarrollado bajo una arquitectura de 3 capas la cual está soportada por los frameworks Hibernate, Spring y Jackson para la conversiones a JSON, el objetivo es exponer un recurso REST el cual pueda ser consumido desde JavaScript (formularios de bonita), Java (conectores de procesos) y aplicaciones nativas de celulares.

Para consumir el servicio desde JavaScript se deben utilizar llamados de tipo XMLHttpRequest y para consumirlo desde Java se debe utilizar el cliente desarrollado por Trycore para tal fin el cual detallaremos más adelante.

A continuación el diagrama de arquitectura:



Operaciones del servicio

En esta sección describiremos todas las funcionalidades que posee el servicio así como la lista de parámetros y retorno que genera cada una, cabe destacar que este servicio está diseñado para trabajar con mensajes JSON por lo que el cuerpo de las solicitudes esperadas y de las respuestas que se retornen tendrán este tipo de contenido.

Las operaciones tienen diferentes tipos de parámetros de acuerdo el tipo de solicitud (GET; POST, PUT, DELETE), algunos parámetros son parte de la URL y están rodeados por llaves los cuales llamaremos PathVariable, luego están los que forman parte del QueryString, los que van luego del signo de interrogación, estos los llamaremos RequestParam para los llamados GET y finalmente están los parámetros en viajan en el body de la solicitud, también los llamaremos RequestParam pero son utilizados para los llamados POST y PUT.

## sendText

Esta operación permite realizar el envió de un correo con contenido de tipo

texto.

**Método**: POST

**Ruta de acceso**: /sendText

**Body:** form-data

**Parámetros**:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| to | Destinatario principal del correo, para enviar a varios repetir el parámetro en la solicitud | String | Si |
| cc | Destinatarios con copia del correo para enviar a varios repetir el parámetro en la solicitud | String | No |
| bcc | Destinatarios con copia oculta del correo, para enviar a varios repetir el parámetro en | String | No |
| subject | Asunto del correo | String | Si |
| body | Cuerp del correo en formato texto | String | Si |
| attachment | Parametro multipart para envio de archivo, para enviar a varios repetir el parámetro en la solicitud | MultipartFile | No |

**Respuesta**: void

## sendHtml

Esta operación permite realizar él envió de un correo con contenido de tipo html.

**Método**: POST

**Ruta de acceso**: /sendHtml

**Body:** form-data

**Parámetros**:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| to | Destinatario principal del correo, para enviar a varios repetir el parámetro en la solicitud | String | Si |
| cc | Destinatarios con copia del correo para enviar a varios repetir el parámetro en la solicitud | String | No |
| bcc | Destinatarios con copia oculta del correo, para enviar a varios repetir el parámetro en | String | No |
| subject | Asunto del correo | String | Si |
| body | Cuerp del correo en formato texto | String | Si |
| attachment | Parametro multipart para envio de archivo, para enviar a varios repetir el parámetro en la solicitud | MultipartFile | No |

**Respuesta**: void

## 

## findByEmail

Realiza la búsqueda de correos enviados a la dirección recibida

**Método**: GET

**Ruta de acceso**: /download?e=…

**Parámetros**:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| e | Dirección de correo por la cual filtrar | String | Si |

**Respuesta**: Lista de objetos de tipo Email con la información existente en la base da datos.

Cliente JAVA

Dado que los servicios son de tipo REST con mensajes JSON se creó un cliente JAVA para consumir los servicios a través HTTPClient.

Para poder utilizar el widget se debe importar la librería JAR llamada **email-client.jar** en el studio de bonita o en el proyecto java que se desee, una vez en el classpath se podrá utilizar la clase llamada **EmailClient** la cual contiene las operaciones que permiten invocar el servicio. A continuación detallaremos las operaciones existentes y los parámetros que espera cada una.

## sendText

Esta operación invoca la operación sendText del servicio detallada anteriormente recibe los siguientes parámetros:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| serviceUri | URI del servicio, se debe indicar protocolo, dns, puerto y contexto (por defecto es email-svc) | String | Si |
| to | Lista de destinatarios principales | List<String> | Si |
| cc | Lista de destinatarios con copia | List<String> | No |
| bcc | Lista de destinatarios con copia oculta | List<String> | No |
| subject | Asunto del correo | String | Si |
| body | Cuerp del correo en formato texto | String | Si |
| attachments | Lista de archivos a enviar como adjuntos | List<File> | No |

**Respuesta:** Sin respuesta

## sendHtml

Esta operación invoca la operación sendHtml del servicio detallada anteriormente recibe los siguientes parámetros:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| serviceUri | URI del servicio, se debe indicar protocolo, dns, puerto y contexto (por defecto es email-svc) | String | Si |
| to | Lista de destinatarios principales | List<String> | Si |
| cc | Lista de destinatarios con copia | List<String> | No |
| bcc | Lista de destinatarios con copia oculta | List<String> | No |
| subject | Asunto del correo | String | Si |
| body | Cuerp del correo en formato texto | String | Si |
| attachments | Lista de archivos a enviar como adjuntos | List<File> | No |

**Respuesta:** Sin respuesta

## sendTextSynchronized

Esta operación invoca la operación sendText del servicio pero de forma sincronizada lo cual genera que la solicitud quede esperando respuesta del servicio recibe los siguientes parámetros:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| serviceUri | URI del servicio, se debe indicar protocolo, dns, puerto y contexto (por defecto es email-svc) | String | Si |
| to | Lista de destinatarios principales | List<String> | Si |
| cc | Lista de destinatarios con copia | List<String> | No |
| bcc | Lista de destinatarios con copia oculta | List<String> | No |
| subject | Asunto del correo | String | Si |
| body | Cuerp del correo en formato texto | String | Si |
| attachments | Lista de archivos a enviar como adjuntos | List<File> | No |

**Respuesta:** Sin respuesta

## sendHtmlSynchronized

Esta operación invoca la operación sendHtml del servicio pero de forma sincronizada lo cual genera que la solicitud quede esperando respuesta del servicio recibe los siguientes parámetros:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| serviceUri | URI del servicio, se debe indicar protocolo, dns, puerto y contexto (por defecto es email-svc) | String | Si |
| to | Lista de destinatarios principales | List<String> | Si |
| cc | Lista de destinatarios con copia | List<String> | No |
| bcc | Lista de destinatarios con copia oculta | List<String> | No |
| subject | Asunto del correo | String | Si |
| body | Cuerp del correo en formato texto | String | Si |
| attachments | Lista de archivos a enviar como adjuntos | List<File> | No |

**Respuesta:** Sin respuesta

## findByEmail

Esta operación invoca la operación findByEmail del servicio pero de forma sincronizada lo cual genera que la solicitud quede esperando respuesta del servicio recibe los siguientes parámetros:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| serviceUri | URI del servicio, se debe indicar protocolo, dns, puerto y contexto (por defecto es email-svc) | String | Si |
| email | Direccion de correo para la cual buscar envios | String | Si |

**Respuesta:** Lista de objetos de tipo Document correspondientes a la dirección de correo enviada.