Creatio Academy

Email Listener

Email Listener synchronization service

Version 8.0

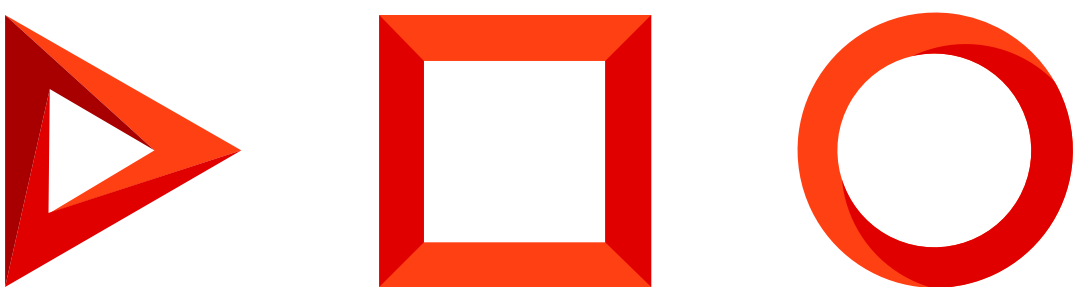


Table of Contents

Email Listener synchronization service

PRODUCTS: ALL CREATIO PRODUCTS

The Email Listener (formerly Exchange Listener) service synchronizes Creatio with Microsoft Exchange and IMAP/SMTP mail services using a subscription mechanism. Email Listener lets you use horizontal scaling that enables the active use of email synchronization block and controlled use of resources.

The synchronization service is required to manage emails in Creatio .NET Framework and .NET Core since version 7.17.2 and .NET 6 since version 8.0.8. This article covers the deployment of Exchange Listener synchronization service for Creatio on-site.

The service consists of the following components:

1. **Email Listener (EL API)**. Initiates an outgoing connection to EWS API or IMAP. Creates a subscription to “new message” events using the mailbox credentials. The open subscription remains in the component memory to ensure fast response time when new emails arrive. The email is downloaded upon receiving the corresponding event. An in-memory repository is sufficient to deploy the service. A required service component.
2. **NoSQL Redis DBMS**. Creates a scalable and fault tolerant system of handler nodes. The Redis repository holds information about the mailboxes that are served. This enables any container to handle Creatio requests to add a new subscription or check the status of a specific subscription regardless of the subscription node. Redis requires a separate database for the Exchange Listener service operation. A required service component.
3. **Email Worker (EL Worker)**. Maintains the scalability and fault tolerance of the primary Email Listener module. The additional module downloads emails from the mail server and delivers them to Creatio. This enables high-load services to handle emails smoother during peaks in the email flow. The EL Worker reduces the load on the EL API components that no longer need to download emails. Instead, the components can manage the subscription and send outgoing emails.
4. **RabbitMQ**. Maintains the scalability and fault tolerance of the service. The queue broker distributes tasks between components in high-load environments.

Determine the configuration of the Email Listener service

Determine the configuration of the Email Listener service for your Creatio instance based on the average flow of emails (both incoming and outgoing) that the company mailboxes handle per second.

For example, if your company uses a single support mailbox whose email flow is 4, the recommended configuration includes 15 EL Worker replicas, 4 EL API replicas, and RabbitMQ service.

Number of mailboxes	Average email flow per second	Number of EL Worker replicas	Number of EL API replicas	RabbitMQ
1-150	<1	0	4	Required
	1-5	15		
	>5	25		
>150	<1	2	1 replica per each 30 mailboxes	
	1-5	15		
	>5	25		

Note. The number of active EL Worker replicas directly affects the email handling speed. The email flow in production fluctuates, therefore certain EL Worker replicas might stand idle for some time. The article provides recommended configuration parameters, but you can use fewer replicas than the table specifies. In this case, the service will take longer to handle the emails during peak load. Optimize the ratio between the email handling speed and the number of resources utilized according to business requirements.

Component replica system requirements

Component	vCPU	RAM
EL Worker	0.1	1.1 GB
EL API	0.150	850 MB
Redis	0.5	3 GB
Rabbit MQ	0.5	4 GB

Note. The values in the table are recommended, the actual resource consumption might vary by the service use case. We recommend monitoring the CPU and memory resources on the deployed services to optimize the available limits.

Email Listener deployment methods

We recommend **using the Kubernetes orchestrator and Helm package manager** to deploy the service. Read more >>>

You can also **use Docker** to speed up the deployment in the development environment. Read more >>>

Deploy the synchronization service via Kubernetes

Deploy the synchronization service using the RabbitMQ programmable message broker.

Take the following steps to deploy the service:

1. Set up the target environment:

a. **Kubernetes cluster**. Learn more about setting up and managing the cluster in the Kubernetes documentation.

b. **Helm package manager**. Learn more about installing the package manager in the Helm documentation.

2. **Install Redis**. Learn more about installing Redis using Helm on the GitHub website.

Example of a command that installs Redis

```
helm install --namespace default --set auth.enabled=true --set auth.password=password --set s
```

Where:

default is the namespace to install Redis.

redis is an arbitrary name for the Redis instance.

3. **Install the RabbitMQ queue broker**. Use the standard bitnami/rabbitmq helm package in the Exchange Listener namespace with the recommended RabbitMQ parameters. Learn more: bitnami/rabbitmq (GitHub).

Note. To specify the RabbitMQ memory limit, use the rabitmqMemoryHighWatermark option. The option is enabled by default in the bitnami/rabbitmq helm package. Calculate the limit using the formula below:
`## rabbitmqMemoryHighWatermark = 0,4 * resources.limits.memory`

Example that installs RabbitMQ with the minimum set of arguments

```
helm install --atomic --namespace exchange-listener --set resources.limits.cpu=500m --set res
```

Create a virtual host and Email Listener user in the installed RabbitMQ instance:

a. Connect to RabbitMQ and run the following command:

```
kubectl exec test-rabbit-rabbitmq-0 -n exchange-listener --stdin --tty shell-demo -- /bin/b
```

b. Create a virtual host:

```
rabbitmqctl add_vhost ExchangeListener
```

c. Create a user and specify the password. For example, “creatio:”

```
rabbitmqctl add_user creatio
```

d. Set up the user permissions to the virtual host:

```
rabbitmqctl set_permissions --vhost ExchangeListener creatio ".*" ".*" ".*"
```

e. Install the Email Listener module. To install the module, download the helm package. View the available parameters of the helm package in the table below.

Attention. You need **repository access** to set up the current version of the Email Listener service. Contact Creatio support to verify your license and gain access to the repository.

For newer **Kubernetes versions**, specify the API version by adding the following parameter: `--set apiVersion=apps/v1`

Example of a command that uses the address and relative path

```
helm upgrade -i

--set ingress.enabled=true

--set ingress.path=<listener_path>

--set ApiUrl=kubernetes

--set apiVersion=apps/v1

--set-string Redis.Connection="<redis_host>\,password=<password>"

--namespace <namespace_name>

--set WorkerReplicaCount=2

--set ReplicaCount=2

--set RabbitMQ.ExchangeName=NewExchange;

--set RabbitMQ.QueueName=NewQueue;

--set RabbitMQ.Host=test-rabbit-rabbitmq;

--set RabbitMQ.HostApi=test-rabbit-rabbitmq;

--set RabbitMQ.HttpPort=15672;

--set RabbitMQ.AmqpPort=5672;

--set RabbitMQ.VirtualHost=ExchangeListener;

 --set RabbitMQ.Login=creatio;
```

```
--set RabbitMQ.Password=creatio; exchangelistener ./home/creatio/exchangelistener-1.0.17.t

--set service.type=NodePort

--set service.nodePort=port
```

Where:

<redis_host> is the Redis server address.

<kubernetes_url> is the Kubernetes URL or IP address.

ReplicaCount is the number of EL API replicas based on the number of mailboxes and average email flow for your company. View the calculation table above.

WorkerReplicaCount is the number of EL Worker replicas based on the number of mailboxes and average email flow for your company. View the calculation table above.

To set up an HTTPS connection, deploy the service with Ingress and a valid SSL certificate, as well as specify HTTPS in the <kubernetes_url> Email Listener service address.

To check the availability, execute the query as specified in the Fig. 1.

```
<kubernetes_url>/<listener_path>/api/listeners/status
```

Example of a command that uses Node IP and port address:

```
helm upgrade -i exchangelistener ./home/creatio/exchangelistener-1.0.17.tgz --namespace de
```

Fig. 1 Email Listener service response


```
{
  "ServiceStatus": "Started",
  "version": "0.5.0",
  "connections": {
    "657b3ea8-477f-419c-a07a-4d4cc2158fc5": {
      "SenderEmailAddress": "[redacted]",
      "BpmUser": "[redacted]",
      "BpmEndpoint": "https://[redacted]/0/ServiceModel/ExchangeListenerService.svc/NewEmail",
      "State": "exists",
      "Id": "657b3ea8-477f-419c-a07a-4d4cc2158fc5",
      "RedisKey": "Subscription_657b3ea8-477f-419c-a07a-4d4cc2158fc5_exchangelistener-api-2",
      "UseFullEmail": true
    },
    "332bcae2-0530-4636-bc09-50a594389f53": {
      "SenderEmailAddress": "[redacted]",
      "BpmUser": "[redacted]",
      "BpmEndpoint": "https://[redacted]/0/ServiceModel/ExchangeListenerService.svc/NewEmail",
      "State": "exists",
      "Id": "332bcae2-0530-4636-bc09-50a594389f53",
      "RedisKey": "Subscription_332bcae2-0530-4636-bc09-50a594389f53_exchangelistener-api-1",
      "UseFullEmail": true
    },
```

Available parameters of the Email Listener helm package

Parameter	Parameter description	Default value
replicaCount	Number of StatefulSet handlers.	2
service.type	Service type. Learn more about the Kubernetes service types in the Kubernetes documentation.	ClusterIP
service.nodePort	If the service.type parameter equals NodePort, specify the external service port in this parameter. Learn more about the NodePort type in the Kubernetes documentation.	
env.host	Host address for Redis.	
env.port	Host port for Redis.	6379
env.base	Database number for Redis.	0.
ingress.enabled	Use address overriding via ingress.	false
ApiUrl	Service address if ingress.enabled=true	
ingress.path	Relative service path.	
log4Net.level	Default logging level.	Info

Use the system requirements calculator to check the server requirements.

Deploy the synchronization service in Docker

To set up the service, use a server (computer or virtual machine) that has Linux or Windows OS installed.

Attention. We recommend deploying the synchronization service in Docker only to the development environment. This method provides a high deployment speed, but does not enable compliance with the requirements of the product environment, namely: function fault tolerance, scaling for the handling of large request volumes, and a unified approach to component management that uses the container orchestration. For the product environment, we strongly recommend using the Kubernetes orchestrator and Helm package manager.

You need **repository access** to set up the current version of the Email Listener service. Contact Creatio support to verify your license and gain access to the repository.

Take the following steps to deploy the service:

a. Set up the Docker container platform first.

   To install Docker Desktop on Windows Server, follow special instructions on Microsoft website.

   To install Docker on Linux, follow the guide in the Docker documentation. To check the installed Docker version, run the following command:

   ```
   docker --version.
   ```

   You can install Docker components using the Docker-Compose instruction file. Learn more about installing Docker-Compose in the Docker documentation.

b. Install and run Email Listener:

   a. Open the directory to deploy Email Listener on the server dedicated for the service.

   b. Download and unpack the archive that contains the setup files to the directory. Download the archive.

   c. Open the / Creatio Email Listener component directory and run the following command:

      ```
      docker-compose up -d
      ```

      The command might take up to several minutes to complete.

c. Make sure the logs contain no errors by running the following command: docker logs ListenerAPI.

d. Check whether the deployment is complete by opening the http://localhost:10000/ URL, where localhost is the URL of the Email Listener server.

Set up the Email Listener service in Creatio

a. Make sure the ExchangeListenerService anonymous service is available at [*Creatio application address*]/0/ServiceModel/ExchangeListenerService.svc (Fig. 2).

Fig. 2 ExchangeListenerService response

Service

This is a Windows© Communication Foundation service.

Metadata publishing for this service is currently disabled.

If you have access to the service, you can enable metadata publishing by completing the following steps to modify your web or application configuration file:

1. Create the following service behavior configuration, or add the <serviceMetadata> element to an existing service behavior configuration:

```
<behaviors>
    <serviceBehaviors>
        <behavior name="MyServiceTypeBehaviors" >
            <serviceMetadata httpGetEnabled="true" />
        </behavior>
    </serviceBehaviors>
</behaviors>
```

2. Add the behavior configuration to the service:

```
<service name="MyNamespace.MyServiceType" behaviorConfiguration="MyServiceTypeBehaviors" >
```

Note: the service name must match the configuration name for the service implementation.

3. Add the following endpoint to your service configuration:

```
<endpoint contract="IMetadataExchange" binding="mexHttpBinding" address="mex" />
```

Note: your service must have an http base address to add this endpoint.

The following is an example service configuration file with metadata publishing enabled:

```
<configuration>
    <system.serviceModel>

        <services>
            <!-- Note: the service name must match the configuration name for the service implementation. -->
            <service name="MyNamespace.MyServiceType" behaviorConfiguration="MyServiceTypeBehaviors" >
                <!-- Add the following endpoint.  -->
                <!-- Note: your service must have an http base address to add this endpoint. -->
                <endpoint contract="IMetadataExchange" binding="mexHttpBinding" address="mex" />
            </service>
        </services>

        <behaviors>
            <serviceBehaviors>
                <behavior name="MyServiceTypeBehaviors" >
                    <!-- Add the following element to your service behavior configuration. -->
                    <serviceMetadata httpGetEnabled="true" />
                </behavior>
            </serviceBehaviors>
        </behaviors>

    </system.serviceModel>
</configuration>
```

For more information on publishing metadata please see the following documentation: http://go.microsoft.com/fwlink/?LinkId=65455.

b. Set the needed system setting values. To do this:

   a. Open the System Designer, e. g., by clicking .

   b. Click “System settings” in the “System setup” block.

   c. Set the values of the following system settings:

      “**ExchangeListenerServiceUri**” (the “ExchangeListenerServiceUri” code). The format of the system setting: [*the service address used at installation*]/api/listeners.

      “**Creatio exchange events endpoint URL**” (the “BpmonlineExchangeEventsEndpointUrl” code). The format of the system setting value: [*the anonymous ExchangeListenerService address*]/NewEmail. For example, https://mycreatio.com/0/ServiceModel/ExchangeListenerService.svc/NewEmail.

Email Listener service diagnostics

The Email Listener service diagnostics page provides tools for troubleshooting the service.

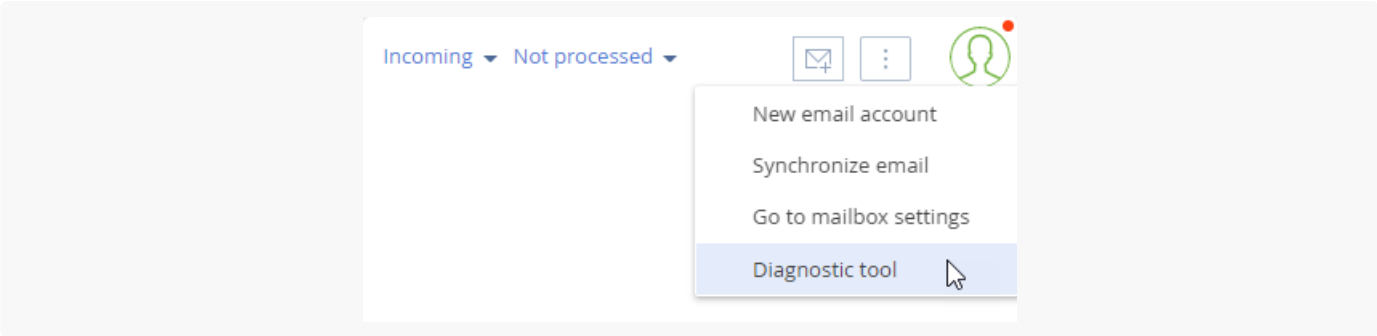
Use the service page:

- to check if the features essential for Exchange Listener are enabled
- to verify the service availability
- to receive subscription information
- to validate the “ExchangeListenerServiceUri” system setting
- to check the health of a mailbox
- to check if the microservice can connect to the Creatio website

You can open the Email Listener service diagnostics page in several ways:

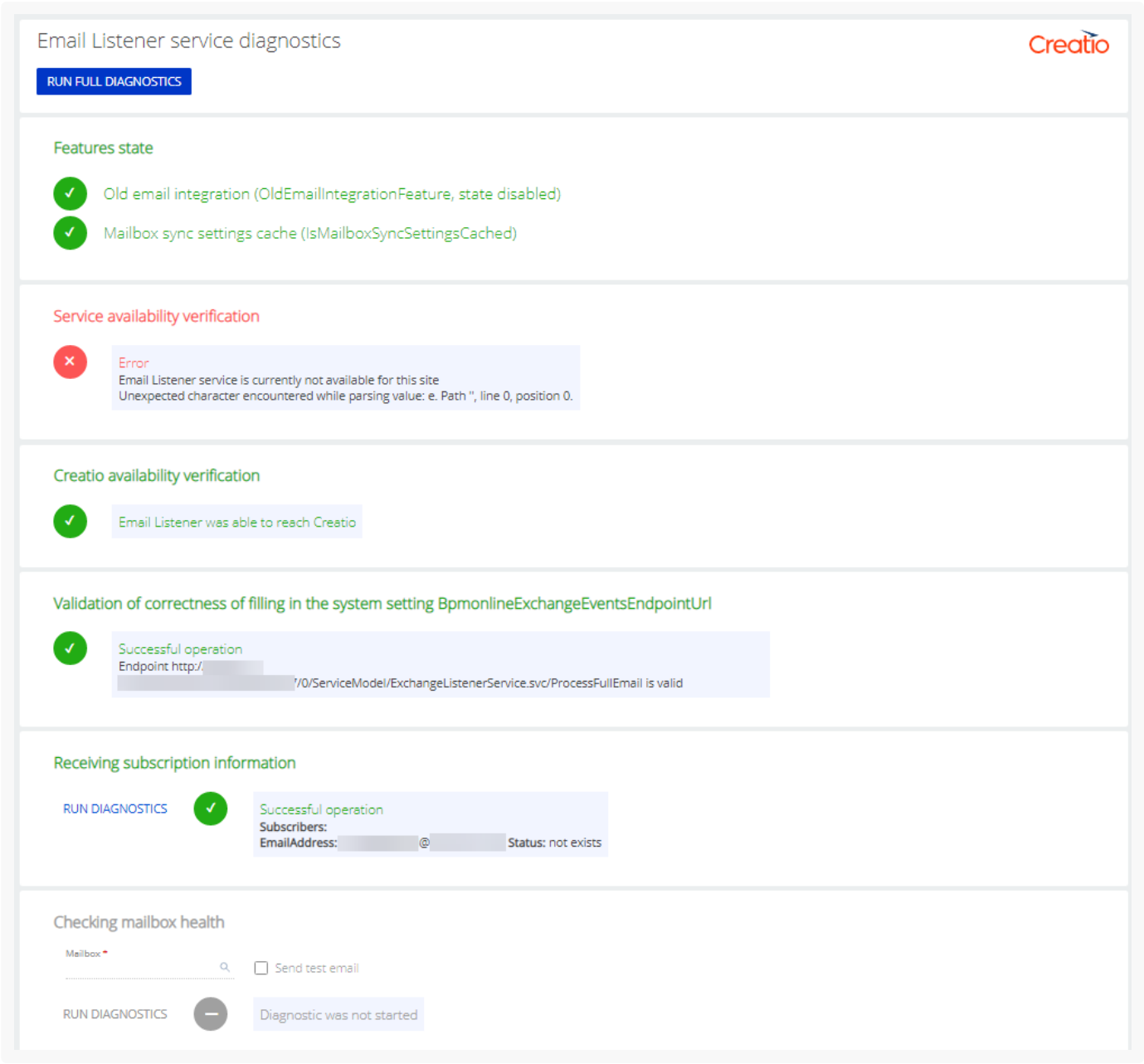
- Use the menu on the Email tab of the communication panel (Fig. 3).
- Use the page of the configured mail services.
- Add the “/0/ClientApp/#/IntegrationDiagnostics/” string to the URL of your Creatio website in the browser address bar and press Enter. For example, `http://mycreatio.com/0/ClientApp/#/IntegrationDiagnostics/ExchangeListener`.

Fig. 3 Open the Email Listener service diagnostics



The diagnostics page contains several readout blocks and diagnostics controls (Fig. 4). By default, most of the readout blocks do not display diagnostics data unless you click [*Run diagnostics*] in that block.

Fig. 4 Email Listener service diagnostics



Feature state	This readout block runs diagnostics automatically on page load. Checks if Email Listener features are enabled in your Creatio application: • ExchangeListenerEnabled • EmailIntegrationV2 • SendEmailsV2 Learn more about enabling features in the developer documentation: Feature Toggle mechanism.
Service availability verification	Checks if the Email Listener service is accessible from your Creatio application.
Receiving subscription information	Checks the connection to the remote server.
Validation of the “ExchangeListenerServiceUri” system setting	Checks if the Exchange Listener service endpoint specified in the “ExchangeListenerServiceUri” system setting is valid.
Checking mailbox health	Checks the operation of Microsoft Exchange mailboxes. Select the [*Send test email*] checkbox to send a test email to the specified address when clicking the [*Run diagnostics*] link.