# Creating Authorization Clients

Release 3.8.0.0
Issue 1
June 2020

**AVAYA SOFTWARE DEVELOPMENT KIT LICENSE AGREEMENT**

**REVISED: October 14, 2019**

READ THIS CAREFULLY BEFORE ELECTRONICALLY ACCESSING OR USING THIS PROPRIETARY PRODUCT!

THIS IS A LEGAL AGREEMENT ("AGREEMENT") BETWEEN YOU, INDIVIDUALLY, AND/OR THE LEGAL ENTITY FOR WHOM YOU ARE OPENING, INSTALLING, DOWNLOADING, COPYING OR OTHERWISE USING THE AVAYA SOFTWARE DEVELOPMENT KIT ("SDK") (COLLECTIVELY, AS REFERENCED HEREIN, "YOU", "YOUR", OR "LICENSEE") AND AVAYA INC. OR ANY AVAYA AFFILIATE (COLLECTIVELY, "AVAYA"). IF YOU ARE ACCEPTING THE TERMS AND CONDITIONS OF THIS AGREEMENT ON BEHALF OF A LEGAL ENTITY, YOU REPRESENT AND WARRANT THAT YOU HAVE FULL LEGAL AUTHORITY TO ACCEPT ON BEHALF OF AND BIND SUCH LEGAL ENTITY TO THIS AGREEMENT. BY OPENING THE MEDIA CONTAINER, BY INSTALLING, DOWNLOADING, COPYING OR OTHERWISE USING THE AVAYA SOFTWARE DEVELOPMENT KIT ("SDK") OR AUTHORIZING OTHERS TO DO SO, YOU SIGNIFY THAT YOU ACCEPT AND AGREE TO BE BOUND BY THE TERMS OF THIS AGREEMENT. IF YOU DO NOT HAVE SUCH AUTHORITY OR DO NOT WISH TO BE BOUND BY THE TERMS OF THIS AGREEMENT, SELECT THE "DECLINE" BUTTON AT THE END OF THE TERMS OF THIS AGREEMENT OR THE EQUIVALENT OPTION AND YOU SHALL HAVE NO RIGHT TO USE THE SDK.

**1.0 DEFINITIONS.**

1.1 "Affiliates" means any entity that is directly or indirectly controlling, controlled by, or under common control with Avaya Inc. For purposes of this definition, "control" means the power to direct the management and policies of such party, directly or indirectly, whether through ownership of voting securities, by contract or otherwise; and the terms "controlling" and "controlled" have meanings correlative to the foregoing.

1.2 "Avaya Software Development Kit" or "SDK" means Avaya technology, which may include Software, Client Libraries, Specification Documents, Software libraries, application programming interfaces ("API"), Software tools, Sample Application Code and Documentation.

1.3 "Client Libraries" mean any enabler code specifically designated as such and included in a SDK. Client Libraries may also be referred to as "DLLs", and represent elements of the SDK required at runtime to communicate with Avaya products or other SDK elements.

1.4 "Change In Control" shall be deemed to have occurred if any person, entity or group comes to own or control, directly or indirectly, beneficially or of record, voting securities (or any other form of controlling interest) which represent more than fifty percent (50%) of the total voting power of the Licensee.

1.5 "Derivative Work(s)" means any translation (including translation into other computer languages), port, compiling of Source Code into object code, combination with a pre-existing work, modification, correction, addition, extension, upgrade, improvement, compilation, abridgment or other form in which an existing work may be recast, transformed or adapted or which would otherwise constitute a derivative work under the United States Copyright Act. Permitted Modifications will be considered Derivative Works.

1.6 "Documentation" includes programmer guides, CDs, manuals, materials, and information appropriate or necessary for use in connection with the SDK. Documentation may be provided in machine-readable, electronic or hard copy form.

1.7 "Intellectual Property" means any and all: (i) rights associated with works of authorship throughout the world, including copyrights, neighboring rights, moral rights, and mask works, (ii) trademark and trade name rights and similar rights, (iii) trade secret rights, (iv) patents, algorithms, designs and other industrial property rights, (v) all other intellectual and industrial property rights (of every kind and nature throughout the world and however designated)

whether arising by operation of law, contract, license, or otherwise, and (vi) all registrations, initial applications, renewals, extensions, continuations, divisions or reissues thereof now or hereafter in force (including any rights in any of the foregoing).

1.8 "Permitted Modification(s)" means Licensee's modifications of the Sample Application Code as needed to create applications, interfaces, workflows or processes for use with Avaya products.

1.9 "Specification Document" means any notes or similar instructions in hard copy or machine readable form, including any technical, interface and/or interoperability specifications that define the requirements and conditions for connection to and/or interoperability with Avaya products, systems and solutions.

1.10 "Source Code" means human readable or high-level statement version of software written in the source language used by programmers and includes one or more programs. Source Code programs may include one or more files, such as user interface markup language (.mxml), action script (.as), precompiled Flash code (.swc), java script (.js), hypertext markup language (.html), active server pages (.asp), C# or C# .Net source code (.cs), java source code (.java), java server pages (.jsp), java archives (.jar), graphic interchange format (.gif), cascading style sheet (.css), audio files (.wav) and extensible markup language (.xml) files.

1.11 "Sample Application Code" means Software provided for the purposes of demonstrating functionality of an Avaya product through the Avaya Software Development Kit.

1.12 "Software" means data or information constituting one or more computer or apparatus programs, including Source Code or in machine-readable, compiled object code form.

**2.0 LICENSE GRANT**.

2.1 SDK License.

A. Provided Licensee pays to Avaya the applicable license fee (if any), Avaya hereby grants Licensee a limited, non-exclusive, non-transferable license (without the right to sublicense, except as set forth in 2.1B(iii)) under the Intellectual Property of Avaya and, if applicable, its licensors and suppliers to (i) use the SDK solely for the purpose of Licensee's internal development efforts to develop applications, interfaces, value-added services and/or solutions, workflows or processes to work in conjunction with Avaya products; (ii) to package Client Libraries for redistribution with Licensee's complementary applications that have been developed using this SDK, subject to the terms and conditions set forth herein; (iii) use Specification Documents solely to enable Licensee's products, services and application solutions to exchange messages and signals with Avaya products, systems and solutions to which the Specification Document(s) apply; (iv) modify and create Derivative Works of the Sample Application Code, Specification Documents and Documentation solely for internal development of applications, interfaces, workflows or processes for use with Avaya products, integration of such applications, interfaces, workflows and processes with Avaya products and interoperability testing of the foregoing with Avaya products; and (v) compile or otherwise prepare for distribution the Sample Application Code with Permitted Modifications, into an object code or other machine-readable program format for distribution and distribute the same subject to the conditions set forth in Section 2.1B.

B. The foregoing license to use Sample Application Code is contingent upon the following: (i) Licensee must ensure that the modifications made to the Sample Application Code as permitted in clause (iv) of Section 2.1A are compatible and/or interoperable with Avaya products and/or integrated therewith, (ii) Licensee may distribute Licensee's application that has been created using this SDK, provided that such distribution is subject to an end user pursuant to Licensee's current end user license agreement ("Licensee EULA") that is consistent with the terms of this Agreement and, if applicable, any other agreement with Avaya (e.g., the Avaya DevConnect Program Agreement), and is equally as protective as Licensee's standard software license terms, but

in no event shall the standard of care be less than a reasonable degree of care, and (iii) Licensee ensures that each end user who receives Client Libraries or Sample Application Code with Permitted Modifications has all necessary licenses for all underlying Avaya products associated with such Client Libraries or Sample Application Code.

Your Licensee EULA must include terms concerning restrictions on use, protection of proprietary rights, disclaimer of warranties, and limitations of liability. You must ensure that Your End Users using applications, interfaces, value-added services and/or solutions, workflows or processes that incorporate the API, Client Libraries, Sample Code or Permitted Modifications adhere to these terms, and You agree to notify Avaya promptly if You become aware of any breach of the terms of Licensee EULA that may impact Avaya. You will take all reasonable precautions to prevent unauthorized access to or use of the SDK and notify Avaya promptly of any such unauthorized access or use.

C. Licensee acknowledges and agrees that it is licensed to use the SDK only in connection with Avaya products (and if applicable, in connection with services provided by or on behalf of Avaya).

D. With respect to Software that contains elements provided by third party suppliers, Licensee may install and use the Software in accordance with the terms and conditions of the applicable license agreements, such as "shrinkwrap" or "click-through" licenses, accompanying or applicable to the Software.

2.2 No Standalone Product. Nothing in this Agreement authorizes or grants Licensee any rights to distribute or otherwise make available to a third party the SDK, in whole or in part, or any Derivative Work in source or object code format on a standalone basis other than the modifications permitted in Section 2.1B of this Agreement.

2.3 Proprietary Notices. Licensee shall not remove any copyright, trade mark or other proprietary notices incorporated in the copies of the SDK, Sample Application Code and redistributable files in Licensee's possession or control or any modifications thereto. Redistributions in binary form or other suitable program format for distribution, to the extent expressly permitted, must also reproduce Avaya's copyright, trademarks or other proprietary notices as incorporated in the SDK in any associated Documentation or "splash screens" that display Licensee copyright notices.

2.4 Third-Party Components. You acknowledge certain software programs or portions thereof included in the SDK may contain software distributed under third party agreements ("Third Party Components"), which may contain terms that expand or limit rights to use certain portions of the SDK ("Third Party Terms"). Information identifying the copyright holders of the Third Party Components and the Third Party Terms that apply is available in the attached Schedule 1 (if any), SDK, Documentation, or on Avaya's web site at: http://support.avaya.com/Copyright (or such successor site as designated by Avaya). The open source software license terms provided as Third Party Terms are consistent with the license rights granted in this Agreement, and may contain additional rights benefiting You, such as modification and distribution of the open source software. The Third Party Terms shall take precedence over this Agreement, solely with respect to the applicable Third Party Components, to the extent that this Agreement imposes greater restrictions on You than the applicable Third Party Terms. Licensee is solely responsible for procuring any necessary licenses for Third Party Components, including payment of licensing royalties or other amounts to third parties, for the use thereof.

2.5 Copies of SDK. Licensee may copy the SDK only as necessary to exercise its rights hereunder.

2.6a No Reverse Engineering. Licensee shall have no rights to any Source Code for any of the software in the SDK, except for the explicit rights to use the Source Code as provided to Licensee hereunder. Licensee agrees that it shall not cause or permit the disassembly, decompilation or reverse engineering of the Software. Notwithstanding the foregoing, if the SDK is rightfully located in a member state of the European Union and Licensee needs information about the Software in the SDK in order to achieve

interoperability of an independently created software program with the Software in the SDK, Licensee will first request such information from Avaya. Avaya may charge Licensee a reasonable fee for the provision of such information. If Avaya refuses to make such information available, then Licensee may take steps, such as reverse assembly or reverse compilation, to the extent necessary solely in order to achieve interoperability of the Software in the SDK with an independently created software program. To the extent that the Licensee is expressly permitted by applicable mandatory law to undertake any of the activities listed in this section, Licensee will not exercise those rights until Licensee has given Avaya twenty (20) days written notice of its intent to exercise any such rights.

2.6.b License Restrictions. To the extent permissible under applicable law, Licensee agrees not to: (i) publish, sell, sublicense, lease, rent, loan, assign, convey or otherwise transfer the SDK; (ii) distribute, disclose or allow use the SDK, in any format, through any timesharing service, service bureau, network or by any other means; (iii) distribute or otherwise use the Software in the SDK in any manner that causes any portion of the Software that is not already subject to an OSS License to become subject to the terms of any OSS License; (iv) link the Source Code for any of the software in the SDK with any software licensed under the Affero General Public License (Affero GPL) v.3 or similar licenses; (v) access information that is solely available to root administrators of the Avaya products, systems, and solutions; (vi) develop applications, interfaces, value-added services and/or solutions, workflows or processes that causes adverse effects to Avaya and third-party products, services, solutions, such as, but not limited to, poor performance, software crashes and cessation of their proper functions; and (vii) develop applications, interfaces, value-added services and/or solutions, workflows or processes that blocks or delays emergency calls; (viii) emulate an Avaya SIP endpoint by form or user interface design confusingly similar as an Avaya product ; (ix) reverse engineer Avaya SIP protocol messages; or (x) permit or encourage any third party to do any of (i) through (x), inclusive, above.

2.7 Responsibility for Development Tools. Licensee acknowledges that effective utilization of the SDK may require the use of a development tool, compiler and other software and technology of third parties, which may be incorporated in the SDK pursuant to Section 2.4. Licensee is solely responsible for procuring such third party software and technology and the necessary licenses, including payment of licensing royalties or other amounts to third parties, for the use thereof.

2.8 U.S. Government End Users. The SDK shall be classified as "commercial computer software" and the Documentation is classified as "commercial computer software documentation" or "commercial items," pursuant to FAR 12.212 or DFAR 227.7202, as applicable. Any use, modification, reproduction, release, performance, display or disclosure of the SDK or Documentation by the Government of the United States shall be governed solely by the terms of the Agreement and shall be prohibited except to the extent expressly permitted by the terms of the Agreement.

2.9 Limitation of Rights. No right is granted to Licensee to sublicense its rights hereunder. All rights not expressly granted are reserved by Avaya or its licensors or suppliers and, except as expressly set forth herein, no license is granted by Avaya or its licensors or suppliers under this Agreement directly, by implication, estoppel or otherwise, under any Intellectual Property right of Avaya or its licensors or suppliers. Nothing herein shall be deemed to authorize Licensee to use Avaya's trademarks or trade names in Licensee's advertising, marketing, promotional, sales or related materials.

2.10 Independent Development.

2.10.1 Licensee understands and agrees that Avaya, Affiliates, or Avaya’s licensees or suppliers may acquire, license, develop for itself or have others develop for it, and market and/or distribute applications, interfaces, value-added services and/or solutions, workflows or processes similar to that which Licensee may develop. Nothing in this Agreement shall restrict or limit the rights of Avaya, Affiliates, or Avaya’s licensees or suppliers to commence or continue with the development or distribution of such applications, interfaces, value-added services and/or solutions, workflows or processes.

2.10.2 Nonassertion by Licensee. Licensee agrees not to assert any Intellectual Property related to the SDK or applications, interfaces, value-added services and/or solutions, workflows or processes developed using the SDK against Avaya, Affiliates, Avaya's licensors or suppliers, distributors, customers, or other licensees of the SDK.

2.11 Feedback and Support. Licensee agrees to provide any information, comments, problem reports, enhancement requests and suggestions regarding the performance of the SDK (collectively, "Feedback") via any public or private support mechanism, forum or process otherwise indicated by Avaya. Avaya monitors applicable mechanisms, forums, or processes but is under no obligation to implement any of Feedback, or be required to respond to any questions asked via the applicable mechanism, forum, or process. Licensee hereby assigns to Avaya all right, title, and interest in and to Feedback provided to Avaya.

2.12(a) Fees and Taxes. To the extent that fees are associated with the license of the SDK, Licensee agrees to pay to Avaya or pay directly to the applicable government or taxing authority, if requested by Avaya, all taxes and charges, including without limitation, penalties and interest, which may be imposed by any federal, state or local governmental or taxing authority arising hereunder excluding, however, all taxes computed upon Avaya's net income. If You move any Software, including the SDK, and as a result of such move, a jurisdiction imposes a duty, tax, levy or fee (including withholding taxes, fees, customs or other duties for the import and export of any such Software), then You are solely liable for, and agree to pay, any such duty, taxes, levy or other fees.

2.12(b) Audit. Avaya shall have the right, at its cost and expense, to inspect and/or audit (i) by remote polling or other reasonable electronic means at any time and (ii) in person during normal business hours and with reasonable notice Licensee's books, records, and accounts, to determine Licensee's compliance with this Agreement. In the event such inspection or audit uncovers non-compliance with this Agreement, then without prejudice to Avaya's termination rights hereunder, Licensee shall promptly pay Avaya any applicable license fees. Licensee agrees to keep a current record of the location of the SDK.

2.13 No Endorsement. Neither the name Avaya, Affiliates nor the names of contributors may be used to endorse or promote products derived from the Avaya SDK without specific prior written permission from Avaya.

2.14 High Risk Activities. The Avaya SDK is not fault-tolerant, and is not designed, manufactured or intended for use or resale as on-line control equipment or in hazardous environments requiring failsafe performance, such as in the operation of nuclear facilities, aircraft navigation or aircraft communications systems, mass transit, air traffic control, medical or direct life support machines, dedicated emergency call handling systems or weapons systems, in which the failure of the Avaya SDK could lead directly to death, personal injury, or severe physical or environmental damage ("high risk activities"). If Licensee uses the Avaya SDK for high risk activities, Licensee does so at Licensee's own risk and Licensee assumes all responsibility and liability for such use to the maximum extent such limitation or exclusion is permitted by applicable law. Licensee agrees that Avaya and its suppliers will not be liable for any claims or damages arising from or related to use of the Avaya SDK for high risk activities to the maximum extent such limitation or exclusion is permitted by law.

2.15 No Virus. Licensee warrants that (i) the applications, interfaces, value-added services and/or solutions, workflows or processes Licensee develops using this SDK will not contain any computer program file that includes time code limitations, disabling devices, or any other mechanism which will prevent the Avaya product (including other software, firmware, hardware), services and networks from being functional at all times (collectively "Time Bombs"); and (ii) the applications, interfaces, value-added services and/or solutions, workflows or processes Licensee develops using this SDK will be free of computer viruses, malicious or other harmful code, black boxes, malware, trapdoors, and other mechanisms which could: a) damage, destroy or adversely affect Avaya product, or services and/or end users; b) allow remote/hidden attacks or access through unauthorized computerized command and

control; c) spy (network sniffers, keyloggers), and d) damage or erase such applications, interfaces, value-added services and/or solutions, workflows or processes developed using this SDK or data, or any computer files or systems of Avaya, Affiliates, and/or end users (collectively "Virus"). In addition to any other remedies permitted in the Agreement, if Licensee breaches its warranties under this Section, Licensee will, at its expense, take remedial action to eliminate any Time Bombs and/or Viruses and prevent re-occurrence (including implementing appropriate processes to prevent further occurrences) as well as provide prompt, reasonable assistance to Avaya to materially reduce the effects of the Time Bomb and/or Virus.

2.16 Disclaimer. Any software security feature is not a guaranty against malicious code, deleterious routines, and other techniques and tools employed by computer "hackers" and other third parties to create security exposures. Compromised passwords represent a major security risk. Avaya encourages You to create strong passwords using three different character types, change Your password regularly and refrain from using the same password regularly. You must treat such information as confidential. You agree to notify Avaya immediately upon becoming aware of any unauthorized use or breach of Your user name, password, account, API Key, or other credentials as provided by Avaya for use of the SDK, or subscription. You are responsible for ensuring that Your networks and systems are adequately secured against unauthorized intrusion or attack and regularly back up of Your data and files in accordance with good computing practices.

2.17 Third Party Licensed Software

A. "Commercial Third Party Licensed Software" is software developed by a business with the purpose of making money from the use of that licensed software. "Freeware Licensed Software" is software which is made available for use, free of charge and for an unlimited time, but is not Open Source Licensed Software. "Open Source Software" or "OSS" is as defined by the Open Source Initiative ("OSI") https://opensource.org/osd and is software licensed under an OSI approved license as set forth at https://opensource.org/licenses/alphabetical (or such successor site as designated by OSI). These are collectively referred to herein as "Third Party Licensed Software".

B. Licensee represents and warrants that Licensee, including any employee, contractor, subcontractor, or consultant engaged by Licensee, is to the Licensee's knowledge, in compliance and will continue to comply with all license obligations for Third Party Licensed Software used in the Licensee application created using the SDK including providing to end users all information required by such licenses as may be necessary. LICENSEE REPRESENTS AND WARRANTS THAT, TO THE LICENSEE'S KNOWLEDGE, THE OPEN SOURCE LICENSED SOFTWARE EMBEDDED IN OR PROVIDED WITH LICENSEE APPLICATION OR SERVICES DOES NOT INCLUDE ANY OPEN SOURCE LICENSED SOFTWARE CONTAINING TERMS REQUIRING ANY INTELLECTUAL PROPERTY OWNED OR LICENSED BY AVAYA OR END USERS TO BE (A) DISCLOSED OR DISTRIBUTED IN SOURCE CODE OR OBJECT CODE FORM; (B) LICENSED FOR THE PURPOSE OF MAKING DERIVATIVE WORKS; OR (C) REDISTRIBUTABLE ON TERMS AND CONDITION NOT AGREED UPON BY AVAYA OR END USERS.

C. Subject to any confidentiality obligations, trade secret or other rights or claims of Licensee suppliers, Licensee will respond to requests from Avaya or end users relating to Third Party Licensed Software associated with Licensee's use of Third Party Licensed Software. Licensee will cooperate in good faith by furnishing the relevant information to Avaya or end users and the requester within two (2) weeks from the time Avaya or end user provided the request to Licensee.

**3. OWNERSHIP.**

3.1 As between Avaya and Licensee, Avaya or its licensors or suppliers shall own and retain all Intellectual Property rights, in and to the SDK and any corrections, bug fixes, enhancements, updates, improvements, or modifications thereto and Licensee hereby irrevocably transfers, conveys and assigns to Avaya, its licensors and its suppliers all of its right, title, and interest therein. Avaya or its licensors or suppliers shall have the exclusive right to apply for or register any patents, mask work rights, copyrights, and such other proprietary protections with respect thereto. Licensee acknowledges that the license granted under this Agreement does not provide Licensee with title or ownership to the SDK, but only a right of limited use under the terms and conditions of this Agreement.

3.2 Grant Back License to Avaya. Licensee hereby grants to Avaya an irrevocable, perpetual, non-exclusive, sublicensable, royalty-free, fully paid up, worldwide license under any and all of Licensee's Intellectual Property rights related to any Permitted Modifications, to (i) use, make, sell, execute, adapt, translate, reproduce, display, perform, prepare derivative works based upon, distribute (internally and externally) and sublicense the Permitted Modifications and their derivative works, and (ii) sublicense others to do any, some, or all of the foregoing.

**4.0 SUPPORT.**

4.1 No Avaya Support. Avaya will not provide any support for the SDK provided under this Agreement or for any Derivative Works, including, without limitation, modifications to the Source Code or applications built by Licensee using the SDK. Avaya shall have no obligation to provide support for the use of the SDK, or Licensee's application, services or solutions which may or may not include redistributable Client Libraries or Sample Application Code, to any third party to whom Licensee delivers such applications, services or solutions. Avaya further will not provide fixes, patches or repairs for any defects that might exist in the SDK or the Sample Application Code provided under this Agreement. In the event that Licensee desires support services for the SDK, and, provided that Avaya offers such support services (in its sole discretion), Licensee will be required to enter into an Avaya DevConnect Program Agreement or other support agreement with Avaya.

4.2 Licensee Obligations. Licensee acknowledges and agrees that it is solely responsible for developing and supporting any applications, interfaces, value-added services and/or solutions, workflows or processes developed under this Agreement, including but not limited to (i) developing, testing and deploying such applications, interfaces, value-added services and/or solutions, workflows or processes; (ii) configuring such applications, interfaces, value-added services and/or solutions, workflows or processes to interface and communicate properly with Avaya products; and (iii) updating and maintaining such applications, interfaces, value-added services and/or solutions, workflows or processes as necessary for continued use with the same or different versions of end user and/or third party licensor products, and Avaya products.

**5.0 CONFIDENTIALITY.**

5.1 Protection of Confidential Information. Licensee acknowledges and agrees that the SDK and any other Avaya technical information obtained by it under this Agreement (collectively, "Confidential Information") is confidential information of Avaya. Licensee shall take all reasonable measures to maintain the confidentiality of the Confidential Information. Licensee further agrees at all times to protect and preserve the SDK in strict confidence in perpetuity, and shall not use such Confidential Information other than as expressly authorized by Avaya under this Agreement, nor shall Licensee disclose any Confidential Information to third parties without Avaya's written consent. Licensee further agrees to immediately 1) cease all use of all Confidential Information (including copies thereof) in Licensee's possession, custody, or control; 2) stop reproducing or distributing the Confidential Information; and 3) destroy the Confidential Information in Licensee's possession or under its control, including Confidential Information on its computers, disks, and other digital storage devices upon termination of this Agreement at any time and for any reason. Upon request, Licensee will certify in writing its compliance with this Section. The obligations of confidentiality shall not apply to information which

(a) has entered the public domain except where such entry is the result of Licensee's breach of this Agreement; (b) prior to disclosure hereunder was already rightfully in Licensee's possession; (c) subsequent to disclosure hereunder is obtained by Licensee on a non-confidential basis from a third party who has the right to disclose such information to the Licensee; (d) is required to be disclosed pursuant to a court order, so long as Avaya is given adequate notice and the ability to challenge such required disclosure.

5.2 Press Releases. Any press release or publication regarding this Agreement is subject to prior written approval of Avaya.

**6.0 NO WARRANTY.**

The SDK and Documentation are provided "AS-IS" without any warranty whatsoever. AVAYA SPECIFICALLY AND EXPRESSLY DISCLAIMS ANY WARRANTIES OR CONDITIONS, STATUTORY OR OTHERWISE, INCLUDING THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NONINFRINGEMENT AND SATISFACTORY QUALITY. AVAYA DOES NOT WARRANT THAT THE SDK AND DOCUMENTATION ARE SUITABLE FOR LICENSEE'S USE, THAT THE SDK OR DOCUMENTATION ARE WITHOUT DEFECT OR ERROR, THAT OPERATION WILL BE UNINTERRUPTED, OR THAT DEFECTS WILL BE CORRECTED. FURTHER, AVAYA MAKES NO WARRANTY REGARDING THE RESULTS OF THE USE OF THE SDK AND DOCUMENTATION. NEITHER AVAYA NOR ITS SUPPLIERS MAKE ANY WARRANTY, EXPRESS OR IMPLIED, THAT THE SDK OR DOCUMENTATION IS SECURE, SECURITY THREATS AND VULNERABILITIES WILL BE DETECTED OR SOFTWARE WILL RENDER AN END USER'S OR LICENSEE'S NETWORK OR PARTICULAR NETWORK ELEMENTS SAFE FROM INTRUSIONS AND OTHER SECURITY BREACHES.

**7.0 CONSEQUENTIAL DAMAGES WAIVER.**

EXCEPT FOR PERSONAL INJURY CLAIMS, AVAYA SHALL NOT BE LIABLE FOR ANY INCIDENTAL, INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH, ARISING OUT OF OR RELATING TO THIS AGREEMENT OR USE OF THE SDK, OR FOR THE LOSS OR CORRUPTION OF DATA, INFORMATION OF ANY KIND, BUSINESS, PROFITS, OR OTHER COMMERCIAL LOSS, HOWEVER CAUSED, AND WHETHER OR NOT AVAYA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

**8.0 LIMITATION OF LIABILITY.**

EXCEPT FOR PERSONAL INJURY CLAIMS, IN NO EVENT SHALL AVAYA'S TOTAL LIABILITY TO LICENSEE IN CONNECTION WITH, ARISING OUT OF OR RELATING TO THIS AGREEMENT EXCEED FIVE HUNDRED DOLLARS ($500). THE PARTIES AGREE THAT THE LIMITATIONS SPECIFIED IN THIS SECTION WILL APPLY EVEN IF ANY LIMITED REMEDY PROVIDED IN THIS AGREEMENT IS FOUND TO HAVE FAILED OF ITS ESSENTIAL PURPOSE.

**9.0 INDEMNIFICATION**.

Licensee shall indemnify and hold harmless Avaya, Affiliates and their respective officers, directors, agents, suppliers, customers and employees "Indemnified Parties") from and against all claims, demand, suit, actions or proceedings ("Claims") and damages, losses, liabilities, costs, expenses, and fees (including fees of attorneys and other professionals) ("Damages") based upon an allegation pertaining to wrongful use, misappropriation, or infringement of a third party's Intellectual Property right arising from or relating to Licensee's use of the SDK, alone or in combination with other software, such as operating systems and codecs, and the, direct or indirect, use, distribution or sale of any software, Derivative Works or other products (including but not limited to applications, interfaces, and application programming interfaces) developed utilizing the SDK.

Licensee shall defend, indemnify and hold harmless the Indemnified Parties from and against all Claims and Damages arising out of or related to: (i) personal injury (including death); (ii) damage to any person or tangible property caused, or alleged to be caused by Licensee or Licensee's application created by using the SDK; (iii) the failure by Licensee or Licensee's application created by using the SDK to comply with

the terms of this Agreement or any applicable laws; (iv) the breach of any representation, or warranty made by Licensee herein; or (v) Licensee's breach of any obligation under the Licensee EULA.

**10.0 TERM AND TERMINATION.**

10.1 This Agreement will continue through December 31st of the current calendar year. The Agreement will automatically renew for one (1) year terms, unless terminated as specified in Section 10.2 or 10.3 below.

10.2 Either party shall have the right to terminate the Agreement, upon thirty (30) days written notice to the other party.

10.3 Notwithstanding language to the contrary, Avaya may terminate this Agreement immediately, upon written notice to Licensee for breach of Section 2 (License Grant), Section 5 (Confidentiality) or Section 12 (Compliance with Laws). Avaya may also terminate this Agreement immediately by giving written notice if a Change In Control should occur or if Licensee becomes insolvent, or voluntary or involuntary proceedings by or against Licensee are instituted in bankruptcy or under any insolvency law, or a receiver or custodian is appointed for Licensee, or proceedings are instituted by or against Licensee for corporate reorganization or the dissolution of Licensee, which proceedings, if involuntary, have not been dismissed within thirty (30) days after the date of filing, or Licensee makes an assignment for the benefit of its creditors, or substantially all of the assets of Licensee are seized or attached and not released within sixty (60) days thereafter, or if Licensee has ceased or threatened to cease to do business in the regular course.

10.4 Upon termination or earlier termination of this Agreement, Licensee will immediately cease a) all uses of the Confidential Information; b) Licensee agrees to destroy all adaptations or copies of the Confidential Information stored in any tangible medium including any document or work containing or derived (in whole or in part) from the Confidential Information, and certify its destruction to Avaya upon termination of this License. Licensee will promptly cease use of, distribution and sales of Licensee products that embody any such Confidential Information, and destroy all Confidential Information belonging to Avaya as well as any materials that embody any such Confidential Information. All licenses granted will terminate.

10.5 The rights and obligations of the parties contained in Sections 2.3, 2.6, 2.7, 2.10, 2.11, 2.12, 3, and 5 through 18 shall survive any expiration or termination of this Agreement.

**11.0 ASSIGNMENT.**

Avaya may assign all or any part of its rights and obligations hereunder. Licensee may not assign this Agreement or any interest or rights granted hereunder to any third party without the prior written consent of Avaya. The term "assign" includes, but is not limited to, any transaction in which there is a Change In Control or reorganization of Licensee pursuant to a merger, sale of assets or stock. This Agreement shall terminate immediately upon occurrence of any prohibited assignment.

**12.0 COMPLIANCE WITH LAWS.**

Licensee shall comply with all applicable laws and regulations, including without limitation those applicable to data privacy, intellectual property, trade secret, fraud, music performance rights and the export or re-export of technology and will not export or re-export the SDK or any other technical information provided under this Agreement in any form in violation of the export control laws of the United States of America and of any other applicable country. For more information on such export laws and regulations, Licensee may refer to the resources provided in the websites maintained by the U.S. Commerce Department, the U.S. State Department and the U.S. Office of Foreign Assets Control.

**13.0 WAIVER.**

The failure to assert any rights under this Agreement, including, but not limited to, the right to terminate in the event of breach or default, will not be deemed to constitute a waiver of the right to enforce each and every provision of this Agreement in accordance with their terms.

**14.0 SEVERABILITY.**

If any provision of this Agreement is determined to be unenforceable or invalid, this Agreement will not be rendered unenforceable or invalid as a whole, and the provision will be changed and interpreted so as to

best accomplish the objectives of the original provision within the limits of applicable law.

## 15.0 GOVERNING LAW AND DISPUTE RESOLUTION.

**15.1 Governing Law**. This Agreement and any dispute, claim or controversy arising out of or relating to this Agreement ("Dispute"), including without limitation the formation, interpretation, breach or termination of this Agreement, or any issue regarding whether a Dispute is subject to arbitration under this Agreement, will be governed by New York State laws, excluding conflict of law principles, and the United Nations Convention on Contracts for the International Sale of Goods.

15.2 Dispute Resolution. Any Dispute will be resolved in accordance with the provisions of this Section 15. The disputing party shall give the other party written notice of the Dispute in accordance with the notice provision of this Agreement. The parties will attempt in good faith to resolve each controversy or claim within 30 days, or such other longer period as the parties may mutually agree, following the delivery of such notice, by negotiations between designated representatives of the parties who have dispute resolution authority.

15.3 Arbitration of Non-US Disputes. If a Dispute that arose anywhere other than in the United States or is based upon an alleged breach committed anywhere other than in the United States cannot be settled under the procedures and within the timeframe set forth in Section 15.2, it will be conclusively determined upon request of either party by a final and binding arbitration proceeding to be held in accordance with the Rules of Arbitration of the International Chamber of Commerce by a single arbitrator appointed by the parties or (failing agreement) by an arbitrator appointed by the President of the International Chamber of Commerce (from time to time), except that if the aggregate claims, cross claims and counterclaims by any one party against the other party exceed One Million US Dollars at the time all claims, including cross claims and counterclaims are filed, the proceeding will be held in accordance with the Rules of Arbitration of the International Chamber of Commerce by a panel of three arbitrator(s) appointed in accordance with the Rules of Arbitration of the International Chamber of Commerce. The arbitration will be conducted in the English language, at a location agreed by the parties or (failing agreement) ordered by the arbitrator(s). The arbitrator(s) will have authority only to award compensatory damages within the scope of the limitations of Section 8 and will not award punitive or exemplary damages. The arbitrator(s) will not have the authority to limit, expand or otherwise modify the terms of this Agreement. The ruling by the arbitrator(s)) will be final and binding on the parties and may be entered in any court having jurisdiction over the parties or any of their assets. The parties will evenly split the cost of the arbitrator(s)' fees, but Avaya and Customer will each bear its own attorneys' fees and other costs associated with the arbitration. The parties, their representatives, other participants and the arbitrator(s) will hold the existence, content and results of the arbitration in strict confidence to the fullest extent permitted by law. Any disclosure of the existence, content and results of the arbitration will be as limited and narrowed as required to comply with the applicable law. By way of illustration, if the applicable law mandates the disclosure of the monetary amount of an arbitration award only, the underlying opinion or rationale for that award may not be disclosed.

15.4 Choice of Forum for US Disputes. If a Dispute by one party against the other that arose in the United States or is based upon an alleged breach committed in the United States cannot be settled under the procedures and within the timeframe set forth in Section 15.2, then either party may bring an action or proceeding solely in either the Supreme Court of the State of New York, New York County, or the United States District Court for the Southern District of New York. Except as otherwise stated in Section 15.3 each party consents to the exclusive jurisdiction of those courts, including their appellate courts, for the purpose of all

actions and proceedings arising out of or relating to this Agreement.

15.5 Injunctive Relief. Nothing in this Agreement will be construed to preclude either party from seeking provisional remedies, including, but not limited to, temporary restraining orders and preliminary injunctions from any court of competent jurisdiction in order to protect its rights, including its rights pending arbitration, at any time. The parties agree that the arbitration provision in Section 15.3 may be enforced by injunction or other equitable order, and no bond or security of any kind will be required with respect to any such injunction or order.

15.6 Time Limit. Actions on Disputes between the parties must be brought in accordance with this Section within 2 years after the cause of action arises.

**16.0 IMPORT/EXPORT CONTROL.**

Licensee is advised that the SDK is of U.S. origin and subject to the U.S. Export Administration Regulations ("EAR"). The SDK also may be subject to applicable local country import/export laws and regulations. Diversion contrary to U.S. and/or applicable local country law and/or regulation is prohibited. Licensee agrees not to directly or indirectly export, re-export, import, download, or transmit the SDK to any country, end user or for any use that is contrary to applicable U.S. and/or local country regulation or statute (including but not limited to those countries embargoed by the U.S. government). Licensee represents that any governmental agency has not issued sanctions against Licensee or otherwise suspended, revoked or denied Licensee's import/export privileges. Licensee agrees not to use or transfer the SDK for any use relating to nuclear, chemical or biological weapons, or missile technology, unless authorized by the U.S. and/or any applicable local government by regulation or specific written license. Additionally, Licensee is advised that the SDK may contain encryption algorithm or source code that may not be exported to government or military end users without a license issued by the U.S. Bureau of Industry and Security and any other country's governmental agencies, where applicable.

**17.0 AGREEMENT IN ENGLISH.**

The parties confirm that it is their wish that the Agreement, as well as all other documents relating hereto, including all notices, have been and shall be drawn up in the English language only. Les parties aux présentes confirment leur volonté que cette convention, de même que tous les documents, y compris tout avis, qui s'y rattachent, soient rédigés en langue anglaise.

**18.0 ENTIRE AGREEMENT.**

This Agreement, its exhibits, schedules and other agreements or documents referenced herein, constitute the full and complete understanding and agreement between the parties and supersede all contemporaneous and prior understandings, agreements and representations relating to the subject matter hereof. No modifications, alterations or amendments shall be effective unless in writing signed by both parties to this Agreement.

**19. REDISTRIBUTABLE CLIENT FILES.**

The list of SDK client files that can be redistributed, if any, are in the SDK in a file called Redistributable.txt.

**Schedule 1 to Avaya SDK License Agreement**
**Third Party Notices**

1. **CODECS**: WITH RESPECT TO ANY CODECS IN THE SDK, YOU ACKNOWLEDGE AND AGREE YOU ARE RESPONSIBLE FOR ANY AND ALL RELATED FEES AND/OR ROYALTIES, IF ANY. IT IS YOUR RESPONSIBILITY TO CHECK.

THE H.264 (AVC) CODEC IS LICENSED UNDER THE AVC PATENT PORTFOLIO LICENSE FOR THE PERSONAL USE OF A CONSUMER OR OTHER USES IN WHICH IT DOES NOT RECEIVE REMUNERATION TO: (I) ENCODE VIDEO IN COMPLIANCE WITH THE AVC STANDARD ("AVC VIDEO") AND/OR (II) DECODE AVC VIDEO THAT WAS ENCODED BY A CONSUMER ENGAGED IN A PERSONAL ACTIVITY AND/OR WAS OBTAINED FROM A VIDEO PROVIDER LICENSED TO PROVIDE AVC VIDEO. NO LICENSE IS GRANTED OR SHALL BE IMPLIED FOR ANY OTHER USE. ADDITIONAL INFORMATION FOR THE H.264 (AVC) CODEC MAY BE OBTAINED FROM MPEG LA, L.L.C. SEE HTTP://WWW.MPEGLA.COM.

**Contents**

# Introduction

The Authorization Service (AS) is an Avaya Breeze® platform snap-in providing OAuth 2.0 authorization capabilities to applications by enabling them to obtain access tokens. OAuth 2.0 defines 4 roles:

- Resource Owner (usually an end user)
- Resource Server (hosts protected resources)
- Client (application making a resource request)
    - The word “Client” is used to describe both server-based applications and applications running on an end-user device.
- Authorization Server (AS). This role has several responsibilities:
    - Authenticating applications. This can generally only be done reliably for server-based applications rather than end-user clients.
    - Authenticating users. This is generally done in conjunction with an enterprise Identity Provider (IdP).
    - Authorizing requests made from Clients to Resource Servers.

The Authorization Server facilitates both authentication and authorization. One reason that it’s called the Authorization Server rather than the Authentication Server is that authentication is not done very frequently (often only at startup). After successfully authenticating itself and (optionally) a user, a client is issued an “access token” which is then presented to the resource servers along with each request. The resource servers needn’t concern themselves with authentication, all they must do is validate the token and ensure that the token indicates that the requested operation may be invoked by the client holding the token.

Snap-ins running on Avaya Breeze may function as an OAuth client, an OAuth resource server, or both. Non-Breeze applications may only act as an OAuth client by issuing web service requests to Breeze-based resources such as Oceana services.

This document describes how an application (a client and/or a resource server) can integrate with the AS to obtain access tokens and enforce authorization. When your application acts as a client, it will authenticate itself and (optionally) a user, then subsequently obtain an access token. The helper libraries and the sample application described are based on Java, however the AS does not pose any restriction on what language an application may be written in. In the last section of this document, details are provided on the HTTP protocol specifics of integration with the AS in order to help developers that are using a language other than Java.

# Overview

OAuth 2.0 defines four grant types, which clients can use to request access tokens. For the current release, the AS supports three of them. The Avaya implementation of the Authorization Service requires the client to authenticate itself with every one of these grant types, even those that *additionally* authenticate a user.

- Client Credentials Grant Type
  - This grant type only authenticates the client (not a user). It is generally useful for server-based applications that are not acting on the behalf of a particular user.
- Authorization Code Grant Type
  - This grant type authenticates both the client and the user. It is a redirect-based flow.
  - Your application does not handle the user's credentials, instead redirecting the user's browser to the AS for validation of credentials.
  - Once the user's credentials have been validated by the AS, AS redirects the browser back to your application with an authorization code, which your application then exchanges for an access token.
  - The benefit of this flow is that it enables SAML-based authentication which might include Multi-Factor Authentication (MFA)
- Resource Owner Password Credentials Grant Type
  - This grant type authenticates both the client and the user.
  - This flow is simpler than the Authorization Code Grant Type flow, but it is more limited as it does not allow for SAML or MFA.

In each of these cases, your application must authenticate itself using PKI. This process is described below in the Client Credentials Grant Type section.

PKI-based authentication is generally more suited to server-based code rather than code running on an end user device. It is rare that applications running on end-user devices are issued unique certificates, and it would be rarer still for an administrator to provision each of these unique certificates as an authorized client. The private key associated with a certificate should always be tightly held and NOT passed to browsers.

The good news is that if you are a Java developer, our sample libraries will make it quite simple to perform these authentication steps. If you are using another technology for your web application such as Ruby on Rails, Node.js or .NET, it should be relatively simple to examine the Java code and derive how to authenticate in your environment.

---

## Client Credentials Grant Type

The Client Credentials grant type is used to obtain access tokens for client applications. When a client makes an access token request, it needs to authenticate itself. The AS requires that the authentication credentials are in the form of JSON Web Tokens (JWTs), and that the JWT is signed by the client itself using its private key. The X.509 certificate of the client is made available to the AS when it is provisioned on System Manager. When a client makes a token

request, the AS authenticates the client by validating the client's JWT credentials using its certificate. Upon successful validation, an access token is granted.

### Authorization Code Grant Type

The Authorization Code grant type is used to obtain access tokens for end-users. Since this is a redirection-based flow, the client must be capable of interacting with the resource owner's user-agent (typically a web browser) and capable of receiving incoming requests (via redirection) from the Authorization Service. This is the preferred flow to be used with web applications because it enables multi-factor authentication and abstracts the client from the details of how authentication is done. Two authentication mechanisms are supported:

In this flow:

- The server-based client redirects the user agent (browser) to the Authorization Endpoint.
- The AS authenticates the user by LDAP/SAML. (see sub-sections below)
- If the authentication succeeds, it generates an authorization code for the user.
- The AS then redirects the browser back to the client application with the authorization code.
- The client application picks the authorization code from the incoming request and invokes the Token Endpoint to exchange the authorization code for an access token for the user.
- The AS first authenticates the client application by validating the client's JWT credentials. Then, it validates the authorization code present in the request and grants a token.

## LDAP

Here, the AS presents the user with a login screen. The user authentication is performed against a data-source configured under System Manager Directory Synchronization (At System Manager > Users > Directory Synchronization).

## SAML

Here, the AS redirects the user to a SAML Identity Provider (IdP) configured on System Manager (At Avaya Breeze® > Configuration > Authorization > Authentication Mechanisms > SAML). The IdP presents a login screen to collect the user's credentials (this could include MFA). On successful authentication, the IdP redirects the browser back to the AS with a SAML assertion. Upon receipt of the SAML assertion, the AS redirects the browser back to the client with an authorization code.

### Resource Owner Password Credentials Grant Type

The OAuth 2.0 spec defines Resource Owner Password Credentials to be suitable when the resource owner (end-user) has a trust relationship with the server-based client. In this case, the end-user passes his/her credentials to the client, and the client makes a token request to the AS, passing these credentials in the same request. The AS authenticates the client by validating the client's JWT credentials. The AS then grants a token if the authentication of the resource owner succeeds. The AS authenticates the user against an LDAP data-source configured under System Manager Directory Synchronization (At System Manager > Users > Directory Synchronization).

# Testing the sample application

The SDK provides a sample application called “TaskList”, which acts as a server-based authorization client. The application demonstrates the use of the Authorization Code grant type, wherein it redirects a user to the AS to get authenticated before he/she can use the application. The sample application must have a Resource that it makes request of. Although non-Breeze Resources are not supported in production, the sample app bundles a Resource for ease of testing.

## Prerequisites

The “TaskList” sample application requires the following components to be installed:

- openssl
- Java keytool (JDK 1.7 +)
- Apache Maven (3.2.1 + )

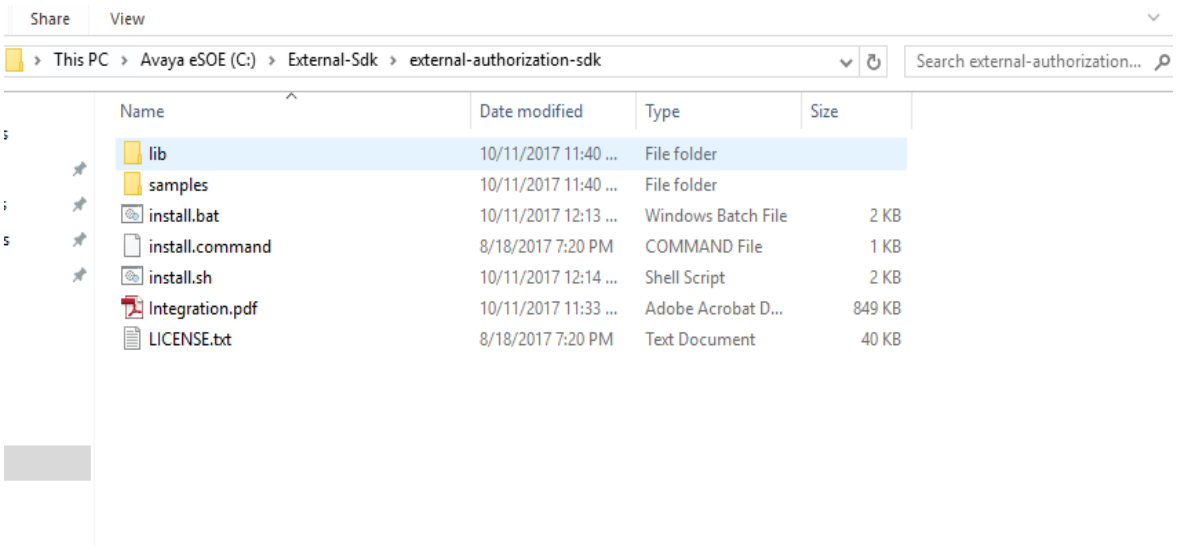
Please note that it is not necessary to have the Avaya Breeze SDK installed on the system where you are running the sample application.

## SDK contents

- lib - authorization-client-helper jar and pom dependencies
- samples/TaskList - TaskList sample application source code
- source - authorization-client-helper library source code
- install files

## Installing the SDK

1. Download the External Authorization SDK and deflate the zip:




2. Execute the install script. The script will install the Authorization Java helper libraries to your local Maven repository.
   Note: The source for sample application "TaskList" is available in the samples folder.

## Configuring certificates

In a production environment, your application will be provisioned with a certificate signed by the System Manager Certificate Authority (CA) or a third-party CA. The sample application, however, will make use of a self-signed certificate which will be installed in the AS's trust store.

This section assumes that all the commands are executed under the directory "**C:\certs**". The files in this folder will again be referenced when the application is being configured to run. If the commands are worked under a different folder, please note to make similar changes in the application configuration. Here are the steps to create a self-signed certificate:

1. Use openssl to generate an example key pair:

   ```
   openssl genrsa -aes256 -out client.key 2048
   ```

2. Generate a CSR using the key created in the above step

   ```
   openssl req -x509 -sha256 -new -key client.key -out client.csr
   ```

   You will be asked to enter a few fields (for creating the CSR and DN), an example is shown below:

```
Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:
```

3. Once the CSR is generated, get it signed by a CA. We are going to self-sign the CSR here:

```
openssl x509 -sha256 -days 3652 -in client.csr -signkey client.key -out client.crt
```

Take a note of client.crt - This certificate will be used further while adding the application as an external client in System Manager.

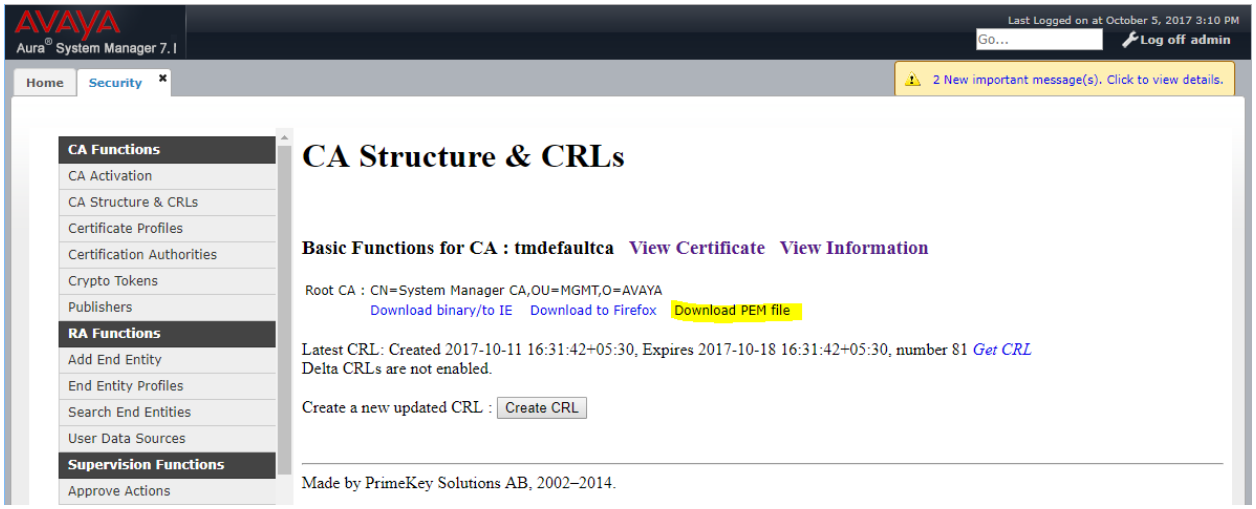
4. Create a PKCS#12 store from the generated key and certificate:

```
openssl pkcs12 -export -name clientcert -in client.crt -inkey client.key -out
keystore.p12
```

5. Convert the PKCS#12 store to a Java KeyStore:

```
keytool -importkeystore -destkeystore clientkeystore.jks -srckeystore
keystore.p12 -srcstoretype pkcs12 -alias clientcert
```

6. Download the System Manager certificate to your local machine. Note that in some cases, a third-party CA may be in use in which case that CA certificate should be installed as trusted rather than the System Manager CA certificate.



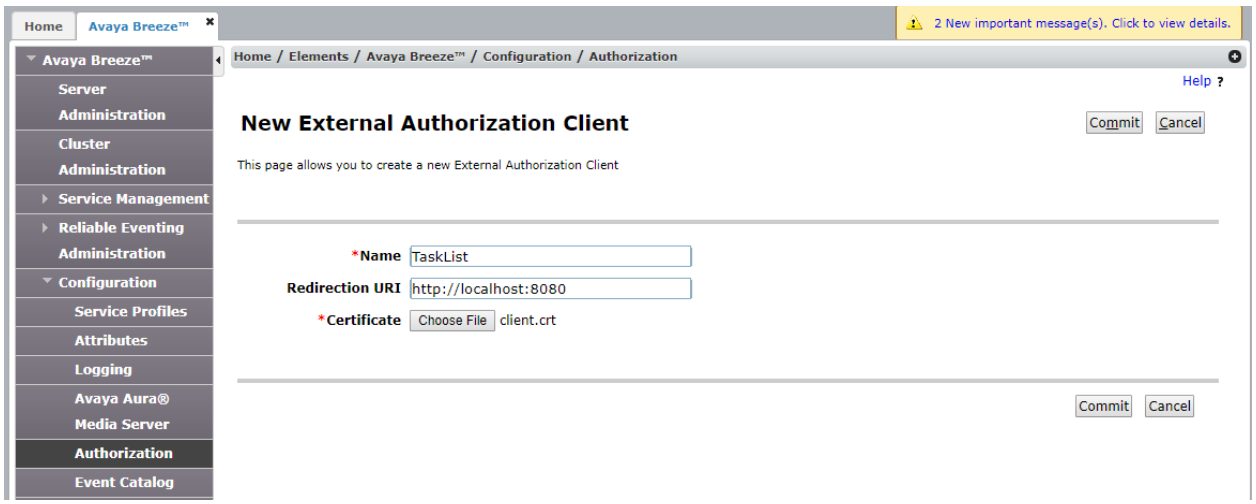
7. Import this certificate into a Java trust store on your local machine:

```
# Import SMGR CA cert onto a trust store
keytool -import -noprompt -alias smgrca -keystore clienttruststore.jks -file
SystemManagerCA.cacert.pem -storepass password
```

Importing the certificate into a trust store is necessary because this trust store will be used by the helper library in the sample application when establishing TLS with the Avaya Breeze cluster for making access token requests. Refer to the section "Initializing the helper" where it is being used.

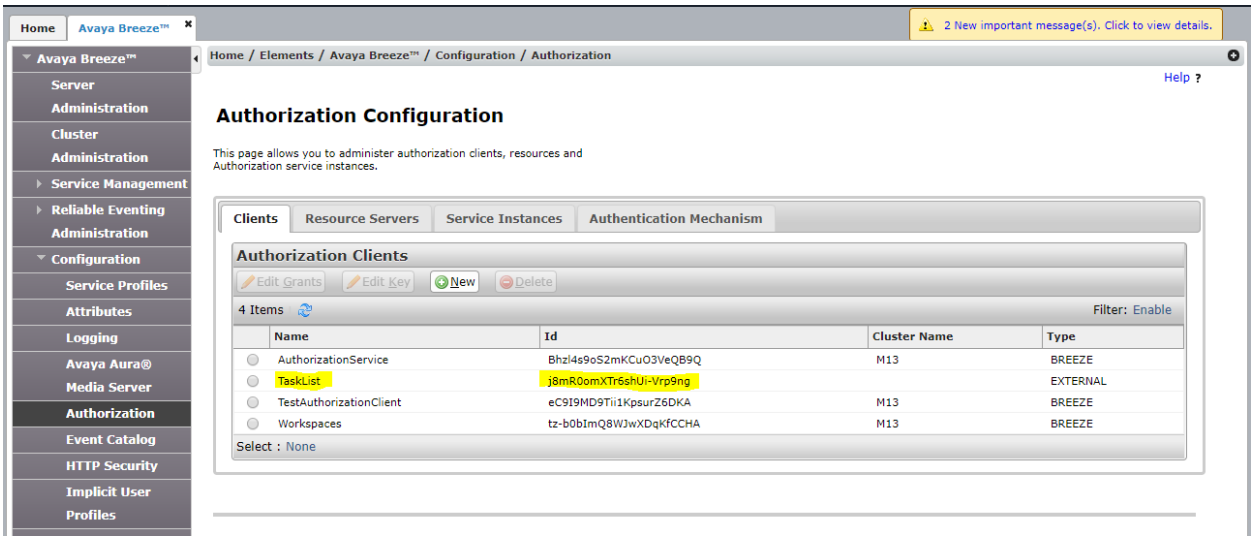
# Configuring the application as an External Authorization Client on System Manager

1. Go to Avaya Breeze® > Configuration > Authorization > Client tab > Select New:




Enter a name for the client (TaskList).
The next field is the URI to which the AS will redirect the browser after authenticating the user. For the sample application considered, since the application runs on your machine, provide the URI as http://localhost:8080.
(For releases older than 3.4, this field isn't available, hence it can be ignored – the AS picks the redirectUri parameter coming from the HTTP request)
Next, click on "Choose File" and select "client.crt" generated in the section Configuring certificates. Click on Commit to save.

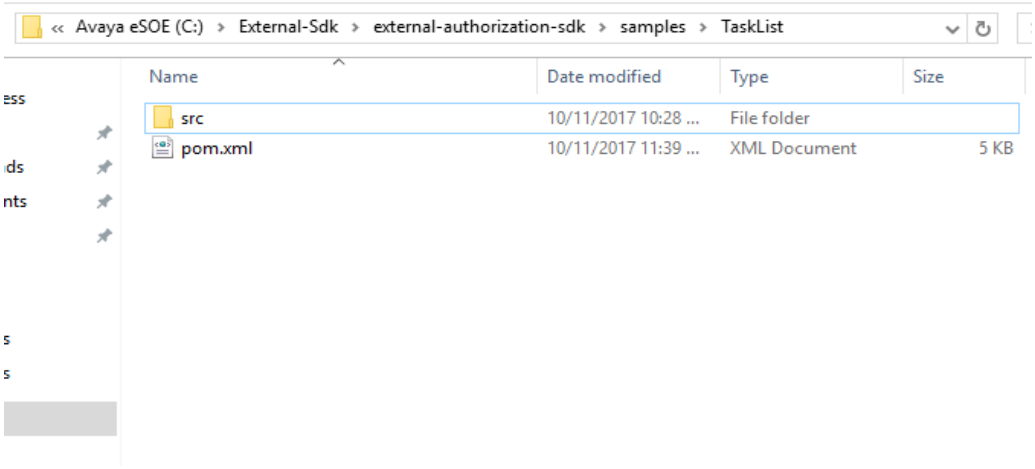
2. This generates a client identifier for the application as shown below:




Take a note of the identifier (in this example, it is j8mR0omXTr6shUi-Vrp9ng). It will be used when the application is being configured to run.

# Running the sample application

1. Navigate to the samples\TaskList folder in deflated SDK:




2. Open the file **application.properties** located under **src\main\resources** folder. Provide values for these properties:

```
spring.jpa.hibernate.ddl-auto=create-drop

# Port on which this application runs
server.port=8080

# URL of the /token endpoint of the AuthorizationService
tokenEndpointURL=https://<FQDN>:9443/services/AuthorizationService/token

# URL of the /authorize endpoint of the AuthorizationService
authorizeEndpointURL=https://<FQDN>:9443/services/AuthorizationService/authorize

# The identifier generated when the application was configured on SMGR
clientId=j8mR0omXTr6shUi-Vrp9ng

# Client keystore properties
clientKeystorePath=C:\\certs\\clientkeystore.jks
clientKeystorePassword=password
clientCertificateAlias=clientcert

# Client truststore properties
clientTruststorePath=C:\\certs\\clienttruststore.jks
clientTruststorePassword=password

sessionCookieName=UserSession
```

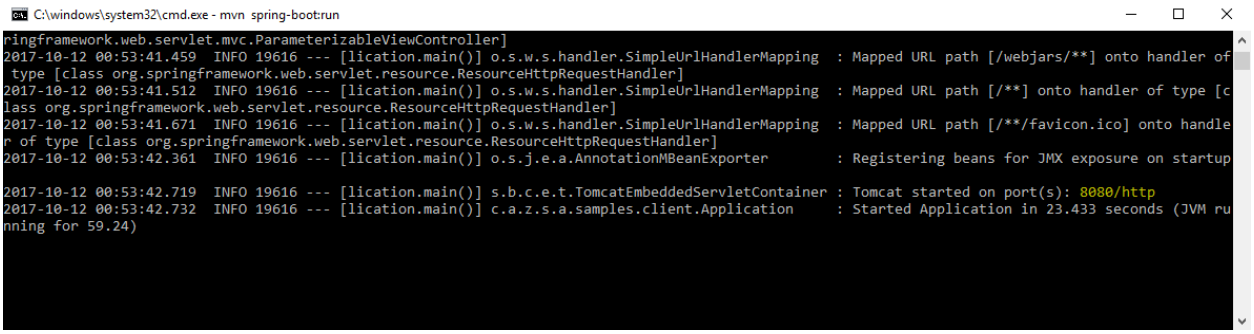
Important notes:

- Values for the tokenEndpointURL, authorizeEndpointURL, clientId will change depending on the environment.
- In a production environment, it would not be recommended to have the keystore / truststore passwords stored in plain text on the filesystem. The sample application does this for simplicity's sake.

3. Open a Command Prompt, navigate to the folder **samples\TaskList** and execute:

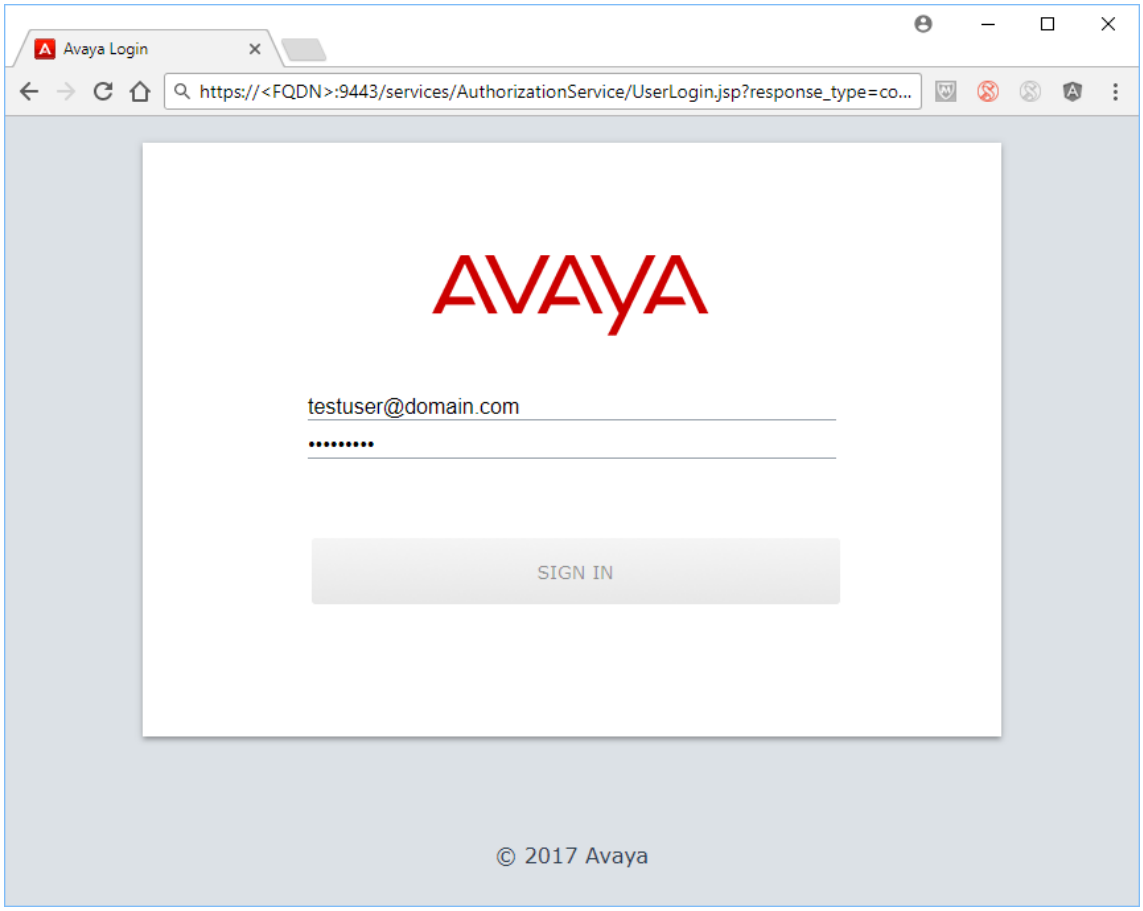
```
mvn spring-boot:run
```

4. Verify the application runs without errors. The logs state that the server has started on port 8080:




5. On a browser, navigate to http://localhost:8080

6. This will redirect to the Authorization Service login screen:

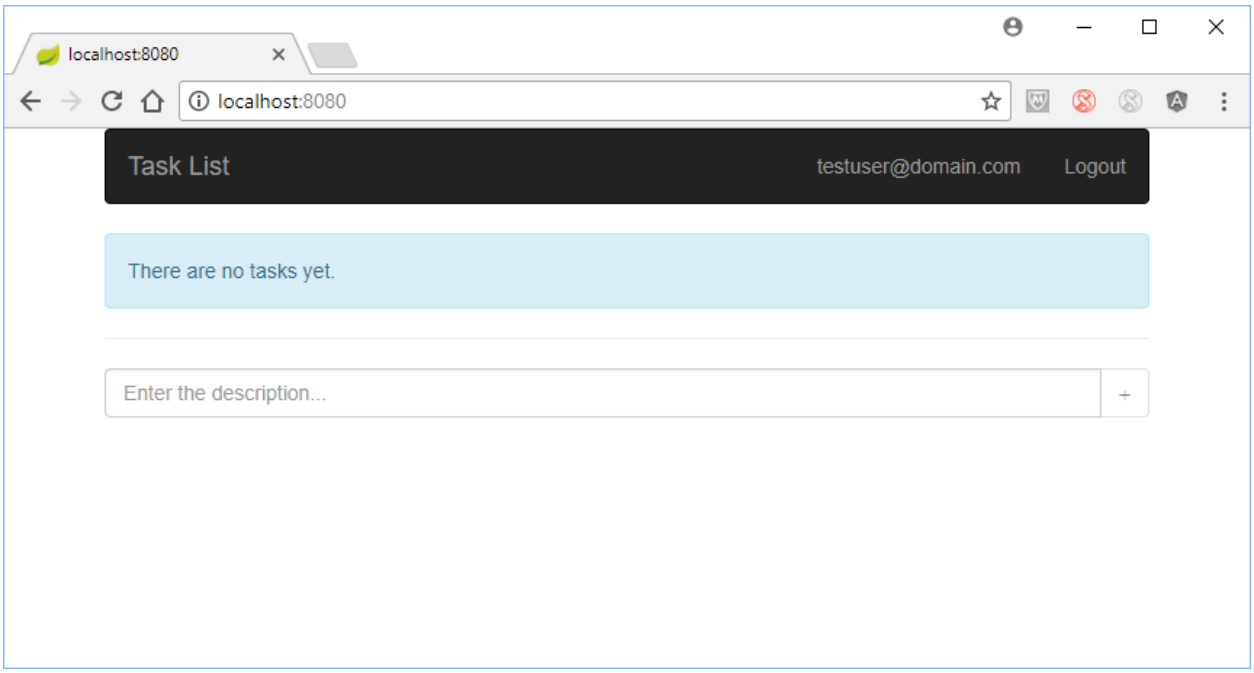



Notes:

- You may see a certificate warning in your browser if the Authorization Service has been deployed on Avaya Breeze servers with certificates that are not fully trusted by your PC. If this is unexpected, contact the Avaya Breeze system administrator.

- If the AS has been configured for SAML-based authentication rather than LDAP, this screen will not be presented. Instead, the AS will redirect a second time to the SAML IdP which will present a login screen.

7. Enter credentials of a user configured on System Manager directory synchronization. Clicking on “SIGN IN” will redirect the user to the application:



# Getting access tokens without Java helper library

The AS provides two endpoints (HTTP resources) to enable authentication and authorization of client applications and end-users. If your client is written in Java, a helper library is provided with the SDK, which masks intricacies of the OAuth 2.0 semantics (explained in this section) and provides Java APIs to get access tokens from the AS. Refer to the section “Java Helper Library” to learn how to use the helper library. If the server portion of your application is not written in Java, read on to learn how to directly interact with the Authorization Service endpoints to obtain access tokens.

## Authorization endpoint

The Authorization endpoint is used only by the Authorization Code grant type. It is used to interact with the end-user (browser) and obtain an authorization grant. Your server-side code will not directly invoke this HTTP endpoint, but rather will redirect the user’s browser to this endpoint when it must authenticate the user. When the user’s browser is redirected to this endpoint, the AS verifies the identity of the user using LDAP or SAML as described in the Overview section. The address for the Authorization endpoint is:

```
https://<FQDN:9443>/services/AuthorizationService/authorize
```

## Token endpoint

The token endpoint is used by the client to get an access token. This endpoint is used by all the three grant types described in the Overview section. Its address is:

```
https://<FQDN:9443>/services/AuthorizationService/token
```

## Client Authentication

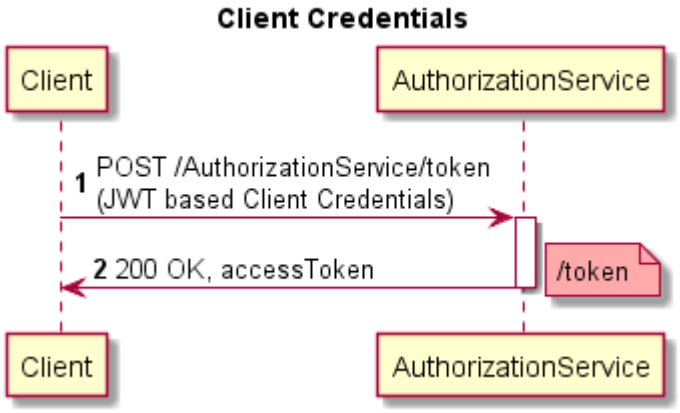
As described in the Overview section, the client application must authenticate itself when making requests to the above endpoints. The authentication itself has two levels:

- Certificate based trust for TLS negotiation - both the Authorization and the Token endpoints serve on the secure port 9443; hence clients must be configured to trust the Avaya Breeze HTTP certificates in order to enable them to establish TLS with the Avaya Breeze cluster when making a request. This is how your client authenticates the Avaya Breeze server hosting the AS.

- JWT based credentials in every HTTP request. This is how the AS authenticates your application. Note that mutual TLS is not required between the client and the AS. Though the AS requires the client certificate to validate its JWT credentials, it is not used during TLS negotiation.

The construction of JWT based credentials depends on which grant type is chosen. The following sections describe this for each of the types supported.

# Client Credentials Grant Type




The flow shown above has the following steps:

1) The client presents its signed JWT when it requests an access token from the token endpoint.
2) The AS authenticates the client by validating its signed JWT. If authentication is successful, the AS then issues an access token.

## JWT Construction

(Part of Step 1)

A JWT contains a header, a payload and a signature. The payload of the JWT contains mandatory "claims" which the AS expects when a client makes a token request. The claims are:

```
"iss" (Issuer) - (A unique identifier for the entity that issued the JWT - Client
identifier)
"sub" (Subject) - (The principal that is the subject of the JWT – the Client
identifier)
"iat" (Issued At) – NumericDate
"jti" (JWT ID) - (JWT Id, a GUID)
```

Use a JWT generation library to complete the JWT construction including the header and the signature. Libraries usually ask for the payload and a private key to generate a signature. Refer to the sample application source for an example. An example JWT looks like this:

```
{
      "iss" : "j8mR0omXTr6shUi-Vrp9ng",
      "sub" : "j8mR0omXTr6shUi-Vrp9ng",
      "iat" : 1455616281,
      "jti" : "3aab05df-ab24-406a-8548-ba32d36271e1"
}
```

Note that the issuer and the subject here are the same. This is because it's the client application which is issuing the JWT. Since the principal who is the subject of the JWT is also the client itself, the subject value must contain an identifier of the client application. This identifier is obtained when the client is provisioned on the System Manager. (See here)

## HTTP Token Request Construction

(Part of Step 1)

The generated JWT is then used in the POST request to the token endpoint as described in the spec:

- "grant_type" parameter must be "client_credentials"
- "client_assertion_type" must be "urn:ietf:params:oauth:client-assertion-type:jwt-bearer".
- "client_assertion" parameter contains the generated JWT.
- The client specifies the scope of the access request using the "scope" request parameter.

Sample HTTP request made by the client using TLS:

```
POST <HOST:9443>/services/AuthorizationService/token HTTP/1.1
Content-Type: application/x-www-form-urlencoded
grant_type=client_credentials&
scope=read&
client_assertion_type=urn%3Aietf%3Aparams%3Aoauth%3Aclient-assertion-
type%3Ajwtbearer&
client_assertion=1uUkSBKFIX7ATndFF5ivnt-8uApHO4k%5B[...].
%5D.hbGciOiJSUzI1NiIsImtpZ%5B[...].
%5D.bUv0XGjQip78AI4z1PrFRNid%5B[...]
```

## Example access token response:

If the access token request is valid and authorized, the AS issues an access token. If the request is poorly formed or if client authentication fails, the authorization server returns an error response as described in the REST APIs section.

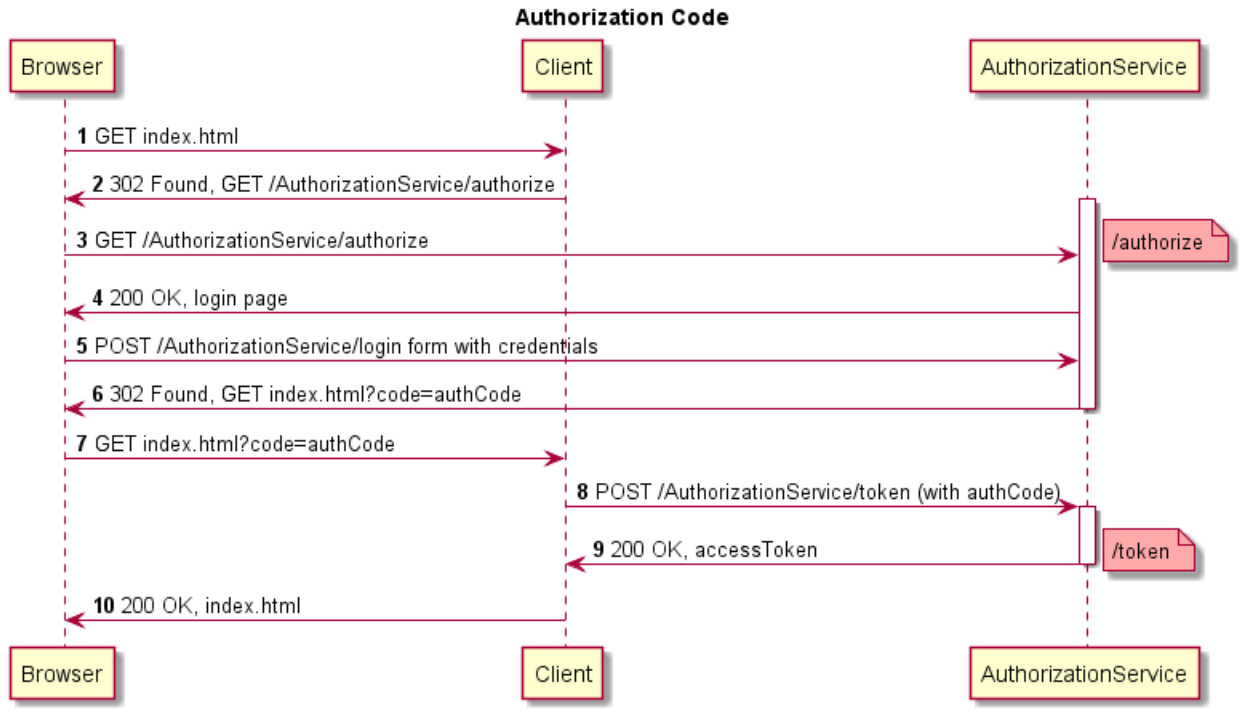
An example successful response:

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8

{
    “access_token” : “1uUkSBKFIX7ATndFF5ivnt-8uApHO4k%5B[...].
                     %5D.hbGciOiJSUzI1NiIsImtpZ%5B[...].
                     %5D.bUv0XGjQip78AI4z1PrFRNid%5B[...]”,
    “expires_in” : 32400,
    “scopes” : [ “read=standard” ],
    “subject” : “j8mR0omXTr6shUi-Vrp9ng”
}
```

Refer to the REST API section for details on the token request and response.

# Authorization Code Grant Type




The flow shown above has the following steps:

1) The end-user accesses the client application.
2) The client redirects the user to the AS.
3) The authorization code request contains the client_id and client redirection_uri.
4) The AS presents the user with a login screen. (in case of SAML, the IdP presents the login screen)
5) The user enters credentials.
6) The AS redirects the user to the client with an authorization code.
7) The client retrieves the authorization code from the request.
8) The client makes an access token request to the AS along with the authorization code.

9) The AS authenticates the client and responds with an access token.
10) The client logs in the user.

## Redirection URI construction

(Step 2)

The redirection is made to the authorization endpoint of the AS:

```
https://<FQDN:9443>/services/AuthorizationService/authorize
```

When the client redirects the end-user to the AS Authorization endpoint (step 3 above), the following query parameters are necessary:

- "response_type", with value as "code"
- "client_id", with the value as the id generated when the client was provisioned on System Manager (see here)
- "redirect_uri", with the value as the client URI to which the AS would redirect the browser along with an authorization code. This URI must match the redirect URI that was provisioned for the client in System Manager.
- "state", with the value as an opaque value used by the client to maintain state between the request and callback.

Sample redirection URI:

```
https://<Cluster_FQDN>:9443/services/AuthorizationService/authorize?
response_type=code&
client_id=j8mR0omXTr6shUi-Vrp9ng&
redirect_uri=http%3A%2F%2Flocalhost%3A8080%2F&
state=7TpGMfQaTUWAdXdbyCZSmQ
```

## Example authorization response

If the end-user authenticates successfully, the AS issues an authorization code and delivers it to the client (step 6 above) by adding the following parameters to the query component of the redirection URI by using the using the "application/x-www-form-urlencoded" format:

- code - The authorization code generated by the AS.
- state - the state parameter that was present in the client authorization request.

A sample redirection response is given below:

```
HTTP/1.1 302 Found
Location: http://localhost:8080/?code=SplxlOBeZQQYbYS6WxSbIA&
state=7TpGMfQaTUWAdXdbyCZSmQ
```

# JWT Construction

(Part of Step 8)

The mandatory “claims” which the AS expects to be present in the payload of the JWT are:

```
"iss" (Issuer) - (A unique identifier for the entity that issued the JWT - Client
identifier)
"sub" (Subject) - (The principal that is the subject of the JWT – the user id)
"iat" (Issued At) – NumericDate
"jti" (JWT ID) - (JWT Id, a GUID)
```

An example JWT looks like this:

```
{
      "iss" : "j8mR0omXTr6shUi-Vrp9ng",
      "sub" : "testuser@domain.com",
      "iat" : 1455616281,
      "jti" : "4dab05df-ac74-406a-8548-ba32d36273f7"
}
```

As shown above, the issuer value is the client identifier (obtained from this section) and the subject value is the user-id. Note that the value in the subject field here is different than what is required when the Client Credentials Grant type is used. Here, the subject of the JWT is user identifier rather than the client identifier used in the Client Credentials Grant Type.

# HTTP Token Request Construction

(Part of Step 8)

This request is made to the token endpoint of the AS:

```
https://<FQDN:9443>/services/AuthorizationService/token
```

The parameters that must be a part of the request are below:

- “grant_type” parameter must be “authorization_code”
- "code" parameter must be the authorization code given by the AS on Step 6.
- "redirect_uri" should be the same as the one included in the initial authorization code request.

The client must also include its JWT credentials (formed in the section JWT Construction) in the Bearer Authorization header of the request.

Sample HTTP request made by the client using TLS:

```
POST <HOST:9443>/services/AuthorizationService/token HTTP/1.1
Authorization: Bearer 1uUkSBKFIX7ATndFF5ivnt-8uApHO4k%5B[...].
%5D.hbGciOiJSUzI1NiIsImtpZ%5B[...].
%5D.bUv0XGjQip78AI4z1PrFRNid%5B[...]
Content-Type: application/x-www-form-urlencoded
grant_type=authorization_code&
code=SplxlOBeZQQYbYS6WxSbIA&
redirect_uri=http%3A%2F%2Flocalhost%3A8080%2F
```

## Example access token response:

If the access token request is valid and authorized, the AS issues an access token. If the request client authentication failed or is invalid, the authorization server returns an error response as described in the REST APIs section.

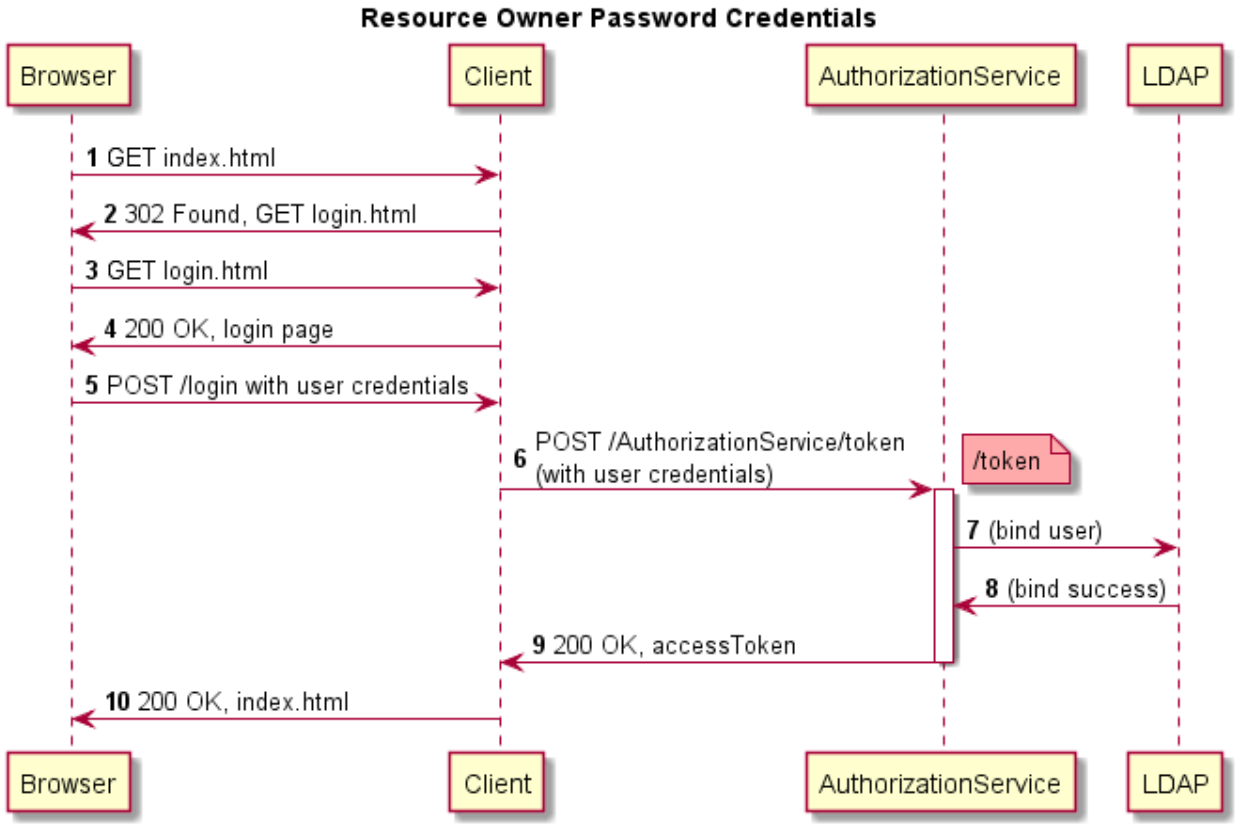
An example successful response:

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8

{
    “access_token” : “1uUkSBKFIX7ATndFF5ivnt-8uApHO4k%5B[...].
                    %5D.hbGciOiJSUzI1NiIsImtpZ%5B[...].
                    %5D.bUv0XGjQip78AI4z1PrFRNid%5B[...]”,
    “expires_in” : 32400,
    “scopes” : [ “ccUserRole=AGENT”, “mappedLdapAttribute” : “upn” ],
    “subject” : “testuser@domain.com”
}
```

Refer to the REST API section for details on the token request and response.

# Resource Owner Password Credentials Grant Type




The flow shown above has the following steps:

1) The end-user accesses the client application.
2) (3 and 4) Client redirects the user to a login page.
5) User enters credentials on the login page.
6) Client makes a token request to the AS along with the user's credentials.
7) (and 8) The AS authenticates the user against the LDAP data source configured under System Manager Data Synchronization.
9) The AS responds to the client with access token.
10) Client logs in the user.

# JWT Construction

(part of Step 6)

The mandatory "claims" which the AS expects to be present in the payload of the JWT are:

```
"iss" (Issuer) - (A unique identifier for the entity that issued the JWT - Client
identifier)
"sub" (Subject) - (The principal that is the subject of the JWT – the user id)
"iat" (Issued At) – NumericDate
"jti" (JWT ID) - (JWT Id, a GUID)
```

An example JWT looks like this:

```
{
      "iss" : "j8mR0omXTr6shUi-Vrp9ng",
      "sub" : "testuser@domain.com",
      "iat" : 1455616281,
      "jti" : "4dab05df-ac74-406a-8548-ba32d36273f7"
}
```

The issuer value is the client identifier (obtained from this section) and the subject value is the user-id.

# HTTP Token Request Construction

(Part of Step 6)

The client then uses the generated JWT (its own credentials) in the Authorization header of the HTTP request.
The user's credentials are then provided in the HTTP request body.

The parameters that must be a part of the request are below:

- "grant_type" parameter must be "password"
- "username" parameter must be the user id.
- "password" parameter must contain the user's password.

The client must also include its JWT credentials (formed in the section JWT Construction) in the Bearer Authorization header of the request.

Sample HTTP request made by the client using TLS:

```
POST <HOST:9443>/services/AuthorizationService/token HTTP/1.1
Authorization: Bearer 1uUkSBKFIX7ATndFF5ivnt-8uApHO4k%5B[...].
%5D.hbGciOiJSUzI1NiIsImtpZ%5B[...].
%5D.bUv0XGjQip78AI4z1PrFRNid%5B[...]
Content-Type: application/x-www-form-urlencoded
grant_type=password&
username=testuser@domain.com&
password=<user_pass>
```

# Example access token response:

If the access token request is valid and authorized, the AS issues an access token. If the request client authentication failed or is invalid, the authorization server returns an error response as described in the REST APIs section.

An example successful response:

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8

{
    “access_token” : “1uUkSBKFIX7ATndFF5ivnt-8uApHO4k%5B[...].
                    %5D.hbGciOiJSUzI1NiIsImtpZ%5B[...].
                    %5D.bUv0XGjQip78AI4z1PrFRNid%5B[...]”,
    “expires_in” : 32400,
    “scopes” : [ “ccUserRole=AGENT”, “mappedLdapAttribute” : “upn” ],
    “subject” : “testuser@domain.com”
}
```

# Java Helper Library

The previous section detailed on how client applications need to construct and make HTTP requests to the AS in order to get access tokens. To summarize, depending on the grant type to be used, the client:

- constructs a JWT
- uses the JWT to construct a token request
- adds grant specific parameters to the token request
- makes the token request to the AS and gets a token

This SDK includes a library that handles all these tasks under the hood and provides Java APIs for each of the grant types supported.

## Importing the library to your project workspace

The SDK install script (refer this section) copies the helper libraries to the system's local maven repository. The library can then be imported onto your Eclipse project by including the dependency:

```
<dependency>
	<groupId>com.avaya.zephyr.sdk.authorization</groupId>
	<artifactId>external-authorization-client-helper</artifactId>
	<version>add-version-here</version>
	<type>jar</type>
</dependency>
```

The artifact version number would be mentioned in the name of the SDK zip file. For example, if the zip file name is: external-authorization-sdk-3.8.0.0.0.zip, then the version tag would look like this: <version>**3.8.0.0.0**</version>

## Required Third Party Maven Dependencies

The library requires two third-party libraries as specified below:

1. org.apache.httpcomponents:httpclient (version 4.5.2 or higher)
2. com.google.guava:guava (version 18.0 or higher)

If the current project already has these dependencies, verify that their versions meet the above requirement. If these dependencies are not present, it is required to explicitly include them, as shown below:

```
<dependency>
	<groupId>org.apache.httpcomponents</groupId>
	<artifactId>httpclient</artifactId>
	<version>4.5.2</version>
</dependency>


<dependency>
	<groupId>com.google.guava</groupId>
	<artifactId>guava</artifactId>
	<version>18.0</version>
</dependency>
```

# Initializing the helper

After importing the library to your project and adding the required dependencies, the helper needs to be initialized. Below code snippet shows how to initialize the helper:

```
// Initialize keystores
KeyStore clientKeyStore = KeyStore.getInstance("JKS");
clientKeyStore.load(new FileInputStream("keystore_location"), "password".
toCharArray());

KeyStore clientTrustStore = KeyStore.getInstance("JKS");
clientTrustStore.load(new FileInputStream("truststore_location"), "password".
toCharArray());

// Initialize the helper
AuthorizationClientHelper clientHelper = new AuthorizationClientHelper.Builder()
.tokenEndpoint("https://<FQDN:9443>/services/AuthorizationService/token")
.clientIdentifier("j8mR0omXTr6shUi-Vrp9ng")
.keyStore(clientKeyStore, "keystore_password", "key_alias")
.trustStore(clientTrustStore)
.build();

// Use the helper to get access tokens
AccessToken accessToken = clientHelper.getAccessToken();
```

There are a few things which this code is doing before making a call to get an access token:

- The first two lines prepare the client keystore. The “keystore_location” is the absolute path of “clientkeystore.jks” which we created in the section Configuring Certificates.
- The next two lines load the client truststore. The “truststore_location” is the absolute path of “clienttruststore.jks” which we created in the section Configuring Certificates.
- The builder requires these arguments (in exact order):

- The Authorization Service token endpoint location
- Client identifier (obtained from this section)
- Keystore (clientKeyStore) initialized on the first two lines with the key alias
- Truststore (clientTrustStore) initialized in the third and fourth lines.

# APIs to get access tokens

The library supports the Client Credentials grant type flow by providing the below two APIs:

```
clientHelper.getAccessToken();
clientHelper.getAccessToken(List<String> scopes)
```

The Authorization Code grant type flow is supported by the below API:

```
clientHelper.getAccessTokenForUser(servletRequest);
```

Resource Owner Password Credentials grant type flow is supported by the following APIs:

```
clientHelper.getAccessTokenForUser(String userName, String userPassword)
clientHelper.getAccessTokenForUser(String userName, String userPassword,List<String>
scopes)
```

Calling shutdown() – The helper API uses this method to clear up system resources when the application is done using the library. This could be a one-time call to be made when the application gets uninstalled. Below code shows this:

```
clientHelper.shutdown();
```

# REST APIs

## Client Credentials Grant Type

**POST *<FQDN>:9443*/services/AuthorizationService/token**

| | |
|---|---|
| URL Query Parameters | None |
| Request Headers | Content-Type: application/x-www-form-urlencoded |
| Request Body | The body is URL-encoded form-data with these fields:<br>grant_type=client_credentials<br>client_assertion_type=urn%3Aietf%3Aparams%3Aoauth%3Aclient-assertion-type%3Ajwt-bearer |

| | |
|---|---|
| | client_assertion=<Client_JWT> |
| Success Response | 200 OK<br>This response body is a JSON, compliant with this JSON schema:<br>`{ "title": "AccessToken", "description": "Token response object from Authorization service", "type": "object", "properties": { "access_token": { "type": "string" }, "expires_in": { "type": "integer" }, "scope": { "type": "string" }, "subject": { "type": "string" } } }` |
| Error Response | 400 Bad Request –<br>• If the request body doesn't contain required parameters<br>• If the client JWT doesn't contain the required claims<br>• Invalid value for client_assertion_type request parameter<br>• Invalid value for grant_type request parameter<br><br>401 Not Authorized –<br>• If the client_assertion request parameter doesn't contain a valid JWT<br>• If the request comes from an un-authorized client.<br><br>403 Forbidden –<br>• If the requested scopes cannot be granted<br>• If the client sends an old JWT<br><br>500 Server Internal Error – Some internal server error |

# Authorization Code Grant Type

**GET *<FQDN>:9443*/services/AuthorizationService/authorize**

| | |
|---|---|
| URL Query Parameters | response_type, client_id, redirect_uri, state |

| | |
|---|---|
| Success Response | 302 Found |
| Error Response | 400 Bad Request –<br>• If the request body doesn't contain required parameters<br><br>401 Not Authorized –<br>• If the request comes from an un-authorized client.<br><br>500 Server Internal Error – Some internal server error |

**POST *<FQDN>:9443*/services/AuthorizationService/token**

| | |
|---|---|
| URL Query Parameters | None |
| Request Headers | Content-Type: application/x-www-form-urlencoded |
| Request Body | The body is URL-encoded form-data with these fields:<br>grant_type= authorization_code<br>code=<auth_code><br>redirect_uri=<client_redirect_uri> |

| | |
|---|---|
| Success Response | 200 OK<br>This response body is a JSON, compliant with this JSON schema:<br>`{`<br>`"title": "AccessToken",`<br>`"description": "Token response object from Authorization service", "type":`<br>`"object",`<br>`"properties": {`<br>`"access_token": {`<br>`"type": "string"`<br>`},`<br>`"expires_in": {`<br>`"type": "integer"`<br>`},`<br>`"scope": {`<br>`"type": "string" },`<br>`"subject": {`<br>`"type": "string"`<br>`}`<br>`}`<br>`}` |
| Error Response | 400 Bad Request –<br>• If the request body doesn't contain required parameters<br>• If the client JWT doesn't contain the required claims<br>• Invalid value for grant_type request parameter<br><br>401 Not Authorized –<br>• If the client_assertion request parameter doesn't contain a valid JWT<br>• If the request comes from an un-authorized client.<br>• Invalid code grant in the request<br><br>403 Forbidden –<br>• If the requested scopes cannot be granted<br>• If the client sends an old JWT<br><br>500 Server Internal Error – Some internal server error |

# Resource Owner Password Credentials Grant Type

**POST *host:9443*/services/AuthorizationService/token**

| | |
|---|---|
| URL Query Parameters | None |
| Request Headers | Content-Type: application/x-www-form-urlencoded<br>`Authorization: Bearer <CLIENT_JWT>` |
| Request Body | The body is URL-encoded form-data with these fields:<br>grant_type=password<br>`username=<username>` |

| | |
|---|---|
| | password=<password> |
| Success Response | 200 OK<br>This response body is a JSON, compliant with this JSON schema:<br>{<br>"title": "AccessToken",<br>"description": "Token response object from Authorization service",<br>"type": "object",<br>"properties": {<br>"access_token": {<br>"type": "string"<br>},<br>"expires_in": {<br>"type": "integer"<br>},<br>"scope": {<br>"type": "string"<br>},<br>"subject": {<br>"type": "string"<br>}<br>}<br>} |
| Error Response | 400 Bad Request –<br>• If the request body doesn't contain required parameters<br>• If the client JWT doesn't contain the required claims<br>• Invalid value for client_assertion_type request parameter<br>• Invalid value for grant_type request parameter<br><br>401 Not Authorized –<br>• If the Authorization Bearer header doesn't contain a valid JWT<br>• If the request comes from an un-authorized client.<br>• If the end user authentication fails. The request is made with wrong credentials (username/password)<br><br>403 Forbidden –<br>• If the requested scopes cannot be granted<br>• If the client sends an old JWT<br><br>500 Server Internal Error – Some internal server error |