고객사

KISCON 등록 업무 자동화

Development Specifications Document (DSD)

Contents

[1. 문서 개요 4](#_Toc7003699)

[2. 프로세스 구성 4](#_Toc7003700)

[3. 프로세스 상세 구조 5](#_Toc7003701)

[3.1 KISCON Automation 5](#_Toc7003702)

[4. 운영 환경 설정 19](#_Toc7003703)

[4.1 로봇 PC 설정 19](#_Toc7003704)

[4.2 오케스트레이터 테넌트 20](#_Toc7003705)

[4.3 로봇 그룹 20](#_Toc7003706)

[4.4 로봇 권한 21](#_Toc7003707)

[4.5 오케스트레이터 asset 22](#_Toc7003708)

[4.6 패키지 등록 23](#_Toc7003709)

[5. 개발자 가이드 24](#_Toc7003710)

[5.1 프로세스 명명 규칙 24](#_Toc7003711)

[5.2 프로젝트 폴더 구조 25](#_Toc7003712)

[5.3 XAML 파일 명명 규칙 25](#_Toc7003713)

[5.4 XAML 파일 테스트 방법 25](#_Toc7003714)

[5.5 Annotation 규칙 25](#_Toc7003715)

[5.6 variable 명명 규칙 26](#_Toc7003716)

[5.7 argument 명명 규칙 26](#_Toc7003717)

[5.8 activity 명명 규칙 26](#_Toc7003718)

[5.9 package version numbering 규칙 27](#_Toc7003719)

[6. 운영 환경 전환 27](#_Toc7003720)

문서 갱신 이력

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 날짜 | 버전 | 성명 | 소속 | 비고 |
| 2019년 4월 24일 | 1.0 | 김형수 | UiPath | 최초 작성 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## 문서 개요

본 문서는 UiPath의 RPA 기술을 이용하여 자동화한 업무의 설계사항을 기술하고, RPA 개발자와 운영자가 운영 및 유지보수에 필요한 제반 정보를 제공한다. RPA 개발자와 운영자는 운영 및 유지보수에서 발생하는 모든 변경 사항을 이 문서에 반영하여 업데이트 해야 한다.

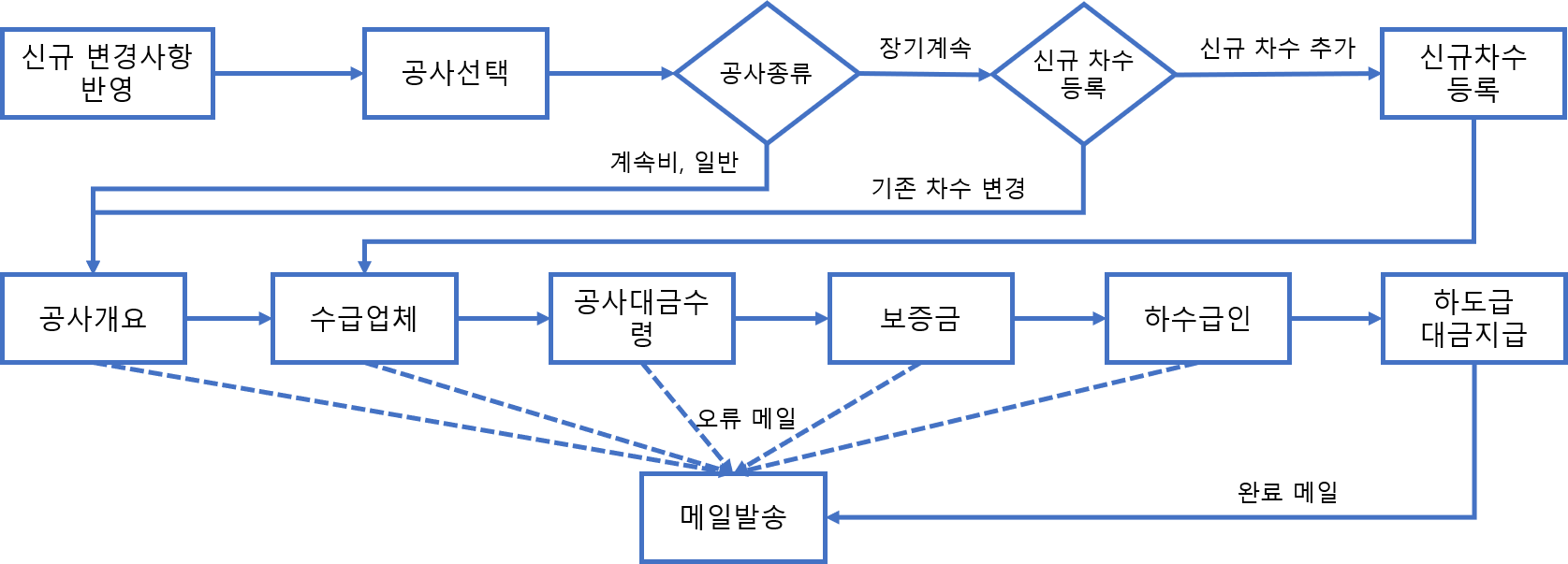
자동화 대상 업무는 KISCON (건설공사정보시스템)에 건설계약 변경사항을 반영하는 하는 것이다. 구체적인 내용은 PDD에 기술한다.

## 프로세스 구성

자동화된 업무는 아래의 1개 프로세스로 구성된다.

|  |  |
| --- | --- |
| 프로세스명 | 역할 |
| KISCON Automation | * 고객사의 DB에서 프로세스 구동 스케줄 정보를 취득 * 고객사 DB에서 KISCON적용 공사 중 변경사항 이 발생한 공사 정보 획득 * 변경된 정보를 기준으로 KISCON 사이트에 접속 후 변경사항 입력 * 변경사항 반영 후 담당자( 운영자, 개별 공사 공무)에게 처리 내용 메일로 통보 |

Orchestrator 스케쥴러가 KISCON Automation프로세스를 구동하면, 가장 먼저 고객사의 DB에서 프로세스 구동 스케줄 정보를 취득하고, 변경사항이 있는지 체크하는 DB 패키지를 호출하며, 변경사항이 있는 공사정보를 획득하며, 변경내용을 KISCON 웹사이트 입력 한다.



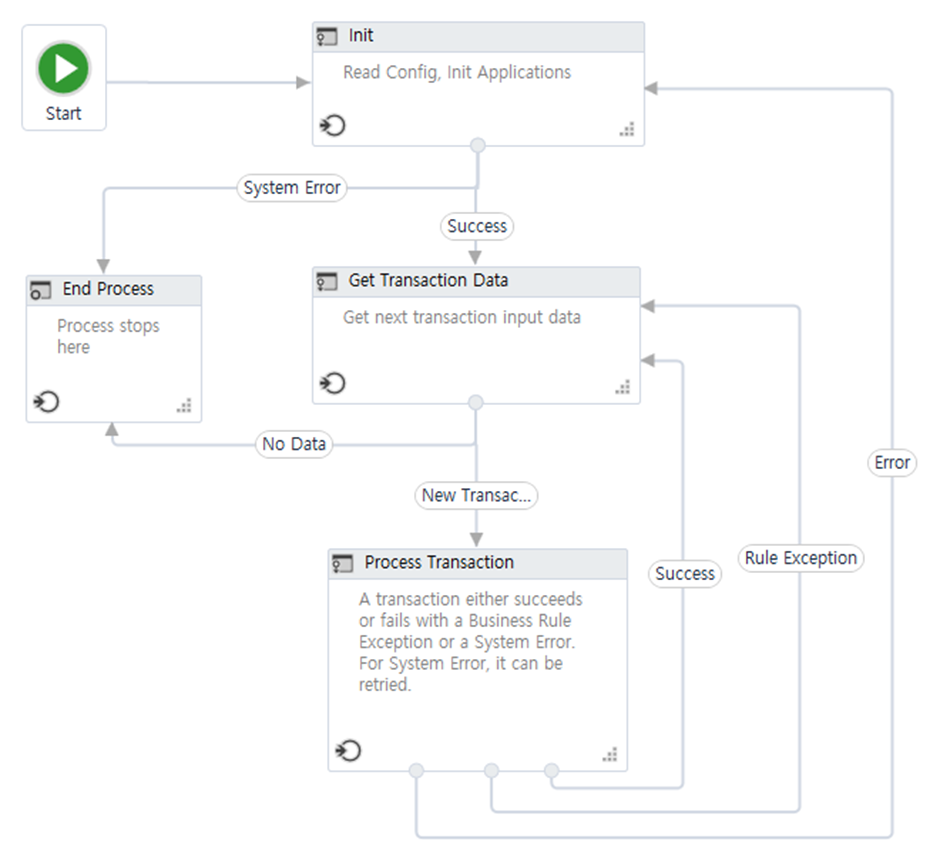
## 프로세스 상세 구조

### KISCON Automation

이 프로세스의 설정 정보는 “Data\Config.xlsx” 파일에 저장되고, 구체적인 내용은 아래와 같다.

|  |  |
| --- | --- |
| Sheet명 | 항목 (설정값) |
| Settings | logField\_BusinessProcessName (TaxInvoice)  : 오케스트레이터 로그에 포함할 프로세스명  DB\_ProviderName (System.Data.OracleClient)  : 데이터베이스 provider 명  DB\_ConnectionString\_\*  : 데이터베이스 connection string  KisconUrl  : KISCON 사이트 접속 URL |
| Constants | BodyTemplate, RowTemplate  : 이메일 본문을 구성하는 데에 사용하는 템플릿들. |
| Assets | BaseRefDate  : 로봇이 자동화를 적용할 기준 일자 ( 기준일자 이후 데이터에 대해서 적용 함) |
| Queries | 고객사 DB에 질의하기 위한 queries. |

이 프로세스의 아키텍쳐는 아래와 같다.



이 프로세스를 구성하는 각각의 state가 수행하는 작업은 아래와 같다.

|  |  |  |
| --- | --- | --- |
| state | 워크플로우 | 수행하는 작업 |
| Init | Framework\InitAllSettings.xaml  Custom\CloseStoppedXPlatform  Framework\KillAllProcesses.xaml  Framework\InitAllApplications | * 프로세스 설정 정보를 메모리에 로딩 * 프로세스에 필요한 애플리케이션 구동 및 초기화 |
| Get Transaction Data | Framework\GetTransactionData | * 고객사 DB에서 처리해야 할 공사정보를 획득 함. |
| Process Transaction | Process.xaml | * 트랜잭션 처리 |
| End Process | Framework\CloseAllApplications  Custom\CheckAndClosePopup | * 사용한 애플리케이션 종료 |

이 프로세스를 구성하는 각 워크플로우(XAML 파일)의 기능은 아래와 같다.

|  |  |
| --- | --- |
| 워크플로우 | 기능 |
| Main.xaml | 프로세스가 시작되는 워크플로우.  Input:  - OrchestratorQueneName: String – 사용하지 않음 |
| Process.xaml | 획득한 공사정보에 대해서 KISCON 사이트에 해당 변경 내용을 업데이트하는 메인 워크플로우  Input  - in\_Config: Dictionary<String, Object> - 설정정보  - in\_TransactionItem : DataRow - 처리할 공사 정보가 저장된 DatRow  - in\_Query: Dictionary<String, Object> - 질의문  - in\_ConnStr: Dictionary<String, Object> - DB설정 정보  - in\_ContinueTab: String - 재시도시 참조할 이전 실행시 완료된 탭 이름  Input/Output  - io\_ErrorList: List<String> - 오류 메세지 리스트  - io\_StatusDict: Dictionary<String,String> - 진행상태 정보 기록하는 맵 |
| Framework\  InitAllSettings.xaml | 프로세스 설정 정보를 로드하는 워크플로우.  Input:  - in\_ConfigFile: String. 설정 파일명.  - in\_ConfigSheets: String[]. sheet 목록.  Output:  - out\_Config: Dictionary<String, Object>. 프로세스 설정 정보.  - out\_Query: Dictionary<String, Object>. 고객사 DB에 대한 DB query strings.  - out\_ConnStr: Dictionary<String, Object>. 고객사 DB에 대한 DB connection strings. |
| Framework\  InitAllApplications.xaml | 프로세스에 필요한 애플리케이션들을 구동하고 초기화함.  Input  - in\_Config: Dictionary<String,Object> - 설정정보  - in\_ConnStr: Dictionarry<String, Object> - DB 연결정보  - in\_Query: Dictionary<String, Object> - 질의문 |
| Framework\  CloseAllApplications.xaml | 사용한 애플리케이션들을 종료함. |
| Framework\  KillAllProcesses.xaml | 정상 종료되지 않은 애플리케이션들을 강제 종료함. |
| Framework\Takesnapshot.xaml | 화면을 캡쳐하여 저장함.  화면을 저장후 7일이 지난 파일은 삭제 함  Input:  - in\_Folder: String. 저장할 폴더.  - io\_FilePath: String. 저장된 파일의 경로명. |
| Framework  SetTransactionStatus.xaml | GetTransactionData를 통해서 획득한 공사정보에 대해서 처리결과를 업데이트한다. 공사정보를 획득하게 되면 UPLOAD\_YN 상태를 P로 변경하고 정상적으로 처리시 UPLOAD\_YN 값을 Y로 수정하게 된다. 만약 시스템 오류로 재시작이 필요한 경우 UPLOAD\_YN을 N으로 변경하게 된다. 지정된 재시도 횟수를 초과하는 경우 UPLOAD\_YN은 P 상태로 남게 되며, GetTransactionData에서 하루가 지나고 UPLOAD\_YN 상태가 P인 항목에 대해서는 N으로 변경후 다시 처리하게 된다.  Input  - in\_Config: Dictionary<String, Object> - 설정정보  - in\_SystemError: Exception - 시스템 에러 정보  - in\_BusinessRuleException: BusinessRuleException - 비지니스 예러 정보  - in\_TransactionItem : DataRow - 처리할 공사 정보가 저장된 DatRow  - in\_TransactionField1: String - 사용하지 않음  - in\_TransactionField2: String - 사용하지 않음  - in\_TransactionID: String - 사용하지 않음  - in\_Query: Dictionary<String, Object> - 질의문  - in\_ConnStr: Dictionary<String, Object> - DB설정 정보  Input/Output  - io\_RetryNumber: Int32 - 재시작 회수  - io\_TransactionNumber: int32 - 트랜잭션 일련번호 |
| Custom\  AddCredentialAssets.xaml | Kiscon 사용자 테이블에서 새로운 사용자가 추가되면 Credential Asset으로 등록하고 PWD는 Null 처리  Input  - in\_KisconUsers: DataTable - 키스콘 사용자 Table  - in\_ConnStr : String - DB연결 정보  - in\_Query: String - 질문의 내용 (패스워드 초기화)  - in\_ProvName: String - DB Client 이름  Output  - out\_UpdateCount: int32 - 패스워드 초기화 된 건수 |
| Custom\  GetAllSubcontractInfo.xaml | KISCON RPA 실행여부를 확인한다  Input:  - in\_ConnStr: String - DB 연결정보  - in\_Query: String - 질문문  - in\_ProvName: String - DB Client 이름  Output  - out\_RunOK:Boolean - 실행할지 말지 여부, True 이면 실행 가능 |
| Custom\  GetherKisconData.xaml | 키스콘 반영을 위한 데이터 처리 (Oracle 패키지)를 호출 한다  Input  - in\_ConnStr: String - DB접속 정보  - in\_Query: String - 질의문  - in\_ProvName: String - DB Client 이름  Output  - 없음. |
| Custom\  GetJobStatus.xaml | 메일을 발송하기 위한 진행상태 정보 조회 및 변환  Input  - in\_ConnStr: String - DB 연결정보  - in\_Query: String - 질문문  - in\_ProjectCode: String - PJT\_CD 내부 관리 데이터 값  - in\_TabKey: String - 해당 트랜젝션 KEY 값  - in\_TabName : String - 마지막 실행한 탭 이름  - in\_ProvName: String - DB Client 이름  Input/ Output  - io\_StatusDict: Dictionary<String,String> - 메일을 보내기 위한 상태 정보 추가. |
| Custom\  GetKisconAccount.xaml | KISCON 사용자(로그인)정보와 연관된 프로젝트를 가져온다. .  Input  - in\_ConnStr: String - DB 연결정보  - in\_Query: String - DB 질의문  - in\_ProjectCode: String - PJT\_CD 값  - in\_ProvName: String - DB Client 이름  Output  - out\_Username: String - Kiscon 사용자ID  - out\_ContractName: String - 기본 계약 명 |
| Custom\  GetKisconAllAcounts.xaml | KISCON 사용자(로그인)정보와 연관된 프로젝트를 가져온다. .  Input  - in\_ConnStr: String - DB 연결정보  - in\_Query: String - DB 질의문  - in\_ProvName: String - DB Client 이름  Output  - out\_ProjectDT: DataTable - KISCON 사용자 정보를 포함한 데이타 테이블 |
| Custom\  GetProjectOverview.xaml | 변경사항이 있는 공사개요 정보를 가져온다.  Input  - in\_ConnStr: String - DB 연결정보  - in\_Query: String - DB 질의문  - in\_ProvName: String - DB Client 이름  Output  - out\_ProjectDT: DataTable - 프로젝트 정보를 포함한 데이타 테이블 |
| Custom\  GetReceiveMoney.xaml | 공사대금 수령 정보를 가져온다.  Input  - in\_ConnStr: String - DB 연결정보  - in\_Query: String - DB 질의문  - in\_ProjectCode: String - 프로젝트(공사명) 코드 PJT\_CD  - in\_ContractSeq: String - 계약 차수 정보  - in\_ProvName: String - DB Client 이름  Output  - out\_ReceivedDT: DataTable - 받은 공사대금 데이터가 있는 |
| Custom\  GetSubcontractInfo.xaml | 하수급인 정보를 조회한다. .  Input  - in\_ConnStr: String - DB 연결정보  - in\_Query: String - DB 질의문  - in\_ProjectCode: String - 프로젝트(공사명) 코드 PJT\_CD  - in\_ProvName: String - DB Client 이름  Output  - out\_SubcontractDT: DataTable - 받은 공사대금 데이터가 있는 테이블 |
| Custom\  GetSubcontractPayment.xaml | 하도급 대금지급(선급금)을 처리한다.  Input  - in\_ConnStr: String - DB 연결정보  - in\_Query: String - DB 질의문  - in\_ProjectCode: String - 프로젝트(공사명) 코드 PJT\_CD  - in\_ProvName: String - DB Client 이름  Output  - out\_PaymentDT: DataTable - 지급할 대금을 포함한 데이블 |
| Custom\  LogKisconStatus.xaml | Kiscon작업 진행사항 로깅  Input  - in\_ConnStr : String - DB연결 정보  - in\_Query: String - 질문의 내용 (상태 로깅)  - in\_Dictionary<String,String> - 로깅 정보 (Key/Value 형태)  필수 key : PJT\_CD, TAB\_NAME, TAB\_KEY, STATUS, MESSAGE  - in\_ProvName: String - DB Client 이름  Output  - 없음 |
| Custom\  SendOutEmail.xaml | 메일을 보내는 기능을 수행함.  Input  - in\_ConnStrDic: Dictionary<String,Object> - DB 연결 정보 데이터  - in\_QueryDict: Dictionary<String,Object> - 쿼리종류  - in\_ConfigDict: Dictionary<String, Object> - 설정 정보  - in\_ProjectCode: String -프로젝트 코드 (공무 담당자에게 메일 발송)  - in\_MainCode: String - 메일 발신 코드 ( CREATE\_MAIL, ERROR\_MAIL)  - in\_Contents: Dictionary<Sting,String> - 보내야 하는 데이터가 있는 내용  - in\_ContractName: String - KISCON 공사명  - in\_ProvName: String - DB Client 이름  Output  - out\_SentCount: int32 - 발송한 메세지 개수 |
| Custom\  UpdateInProgressStatus.xaml | 해당 프로젝트(공사) 작업 진행사 상태를 P로 변경후 시스템/비지니스 오류로 인해서 P상태가 유지되는 경우 +1일 이후에 다시 N으로 변경해주는 처리가 필요함  Input  - in\_ConnStrDict: Dictionary<String,Object> - DB접속 정보  - in\_QueryDict: Dictionary<String,Object> - 질의문 정보  - in\_ProvName: String - DB Client 이름  Output:  - out\_ChangedRows: Int32 - 변경된 행 건수 |
| Custom\  UpdateReceivedMoneyr.xaml | 변경처리 작업은 진행하고 있는 프로젝트의 상태를 변경한다.  N는 처리전, Y는 처리완료, P는 처리중을 의미함  Input  - in\_ConnStr: String - DB 연결정보  - in\_Query: String - DB 질의문  - in\_ProjectCode: String - 프로젝트(공사명) 코드 PJT\_CD  - in\_ContractSeq: String - 계약차수 정보  - in\_ChangeSeq: String - 변경회수 (필요한지 확인 필요)  - in\_Status: String - 처리상태 Y, N, P중 하나  - in\_ProvName: String - DB Client 이름  Output  - out\_Rows: Int32 - 변경된 열 개수 |
| Custom\  UpdateReceivedMoney.xaml | 입력한 공사대금 수령에 대한 업데이트 여부를 표시한다.  Input  - in\_ConnStr: String - DB 연결정보  - in\_Query: String - DB 질의문  - in\_ProjectCode: String - 프로젝트(공사명) 코드 PJT\_CD  - in\_ContractSeq: String - 계약차수정보  - in\_EvalReqNo: String - 공사대금 수령 일련번호  - in\_ProvName: String - DB Client 이름  Output  - out\_Rows: DataRow - 변경사항 정보 |
| Custom\  UpdateSubcontractInfo.xaml | 입력한 하수급인 정보에 대한 업데이트 여부를 표시한다.  Input  - in\_ConnStr: String - DB 연결정보  - in\_Query: String - DB 질의문  - in\_ProjectCode: String - 프로젝트(공사명) 코드 PJT\_CD  - in\_ContractSeq: String - 계약차수 정보  - in\_Brno: String - 하청업체 사업자등록번호  - in\_ProvName: String - DB Client 이름  Output  - out\_Rows: int32 - 변경된 row 개수 |
| Custom\  UpdateSubcontractPayment.xaml | 입력한 하도급 대금지금(기성 및 선급금 )에 대한 업데이트를 처리한다.  Input  - in\_ConnStr: String - DB 연결정보  - in\_Query: String - DB 질의문  - in\_ProjectCode: String - 프로젝트(공사명) 코드 PJT\_CD  - in\_OrdCode: String - 발주번호- 특정회사 지정  - in\_PaymentDate : String - 지급일자 (yyyy-mm-dd 포맷)  - in\_ProvName: String - DB Client 이름  Output  - out\_Rows : Int 32 - 변경된 로우 개수 |
| KISCON\  AddContractSeq.xaml | 새로운 차수 계약을 추가하는 기능을 수행  새로운 차수를 추가 하기 위해서는 현재 최종 차수를 가져와서 비교한 다음에 차수가 없는 경우 새로운 차수로 입력 한다.  Input  - in\_ContractSeq: String - 새로운 차수  - in\_ContractName: String - 공사(계약) 이름  - in\_ProvName: String - DB Client 이름  Output  - 없음 |
| KISCON\  AddNewContractName.xaml | 차수 변경인 경우 새로운 차수 공사 개요를 입력해야 한다.  Input  - in\_IsPublic: Boolean - 공공공사 여부, 아니면 민간공사  - in\_ContractName: String - 공사명(회차 정보 없음)  - in\_ContractSeq: String - 회차 정보  Output  - out\_ContractName: String - 등록한 공사명 |
| KISCON\  AddSubcontractPayment.xaml | 하도급업체 대금지급을 처리함  기 지금내역을과 추가 해야 할 내역을 비교해서 항목에 없는 내용만 처리 함  Input  - in\_PaymentDT: DataTable - 하도급업체 기성/선급금 지급 내역  Output  - 없음 |
| KISCON\  CheckContractSeq.xaml | KISCON에 등록된 차수를 확인 함.  이후 새로운 차수 계약을 할지, 해당 차수 변경 계약을 할지 판단 함.  Input  - in\_ContractName: String - 공사(계약) 이름  Output  - out\_LastContractSeq: String - KISCON에 있는 마지막 차수 계약 정보 |
| KISCON\  GetSelectOptions.xaml | 지정된 Select Element에서 OPTION 항목을 추풀 함  Input  - in\_SelectElement: UiElement - Select 객체  Output  - out\_Options: List<String>: 값을 포함한 리스트 |
| KISCON\  LookupExistingSubcontract.xaml | 신규 계약이 있는지 찾아줌  Input  - in\_CompanyName: String - 업체 명  - in\_ContractDate: String - 계약 체결일  - in\_Existing: DataTable - 기존 계약 체결 정보  Output  - out\_MatchFound: Boolean - 일치하는 정보 여부  - out\_MatchIndex: Int32 - 일치하는 경우 해당 Row 인덱스 (맞는게 없는 경우 -1) |
| KISCON\  LookupExistingSubcontract-  WithAmount.xaml | DB에 있는 내용과 KISCON 화면에 있는 계약이 같은게 있는지 비교, 회사명 + 계약금액 기준  Input  - in\_CompanyName: String - 업체 명  - in\_ContractAmount: String - 계약금액(하도급금액)  - in\_Existing: DataTable - 기존 계약 체결 정보  Output  - out\_MatchFound: Boolean - 일치하는 정보 여부  - out\_RowIndex: Int32 - 일치하는 RowIndex 값 ( 없으면 -1) |
| KISCON\  LookupPaymentHistory.xaml | 대금지급금이 이미 지급되었는지 비교  하도급업체 비교시 (주) 와 주식회사를 동일하게 비교, KISCON에 등록된 업체명 기준으로 변환 후 비교  새롭게 추가시 추가된 내용을 History에 반영한다.  Input  - in\_NormalizedCompanyName: String - KISCON 하도급업체 명  - in\_MoneyType: String - 기성금/선급금 구분  - in\_PayDate:String - 지급일 (yyyy-mm-dd 포맷)  - in\_PayAmount: String - 지급액 (숫자만, 컴마 및 원 없어야 함)  Input/Output  - io\_History: DataTable - 기존 지급 내역  Output  - out\_Found: Boolean - 지급 여부, 기지급된 항목이면 True, 새로운 항목이면 False |
| KISCON\  LookupReceivedMoneyTable.xaml | 기 수령한 금액과 수령예정인 금액을 비고해서 맞는 항목의 row를 넘겨준다.  Input  - in\_ReceivedInfo: DataTable - 수령예정인 금액 테이블  - in\_LookupDate: String - 비교할 수령일자  - in\_LookupAmount: String - 비교할 금액  - in\_LookupType: String - 수령금액 구분(기성/선급금)  Output  - out\_RowIndex: Int32 - 찾은 row 이덱스 없는 경우 -1 |
| KISCON\  LookupSubcontractGuarantee.xaml | 보증서 검색 후 해당 보증서가 이미 등록되어 있는지 파악, 없다면 보증서 추가  Input  - in\_Issuer: String - 대상업체  - in\_IssueNumber: String - 보증서 번호  - in\_IssueDate: String - 발급일자  - in\_Authority: String - 발급기관  - in\_Creditor: String - 보증채권자  Input/Outpt  - io\_TargetDT: DataTable - 기존 보증서 목록  Output  - out\_Found: Boolean - 동일한 보증서가 있다면 True, 없다면 False |
| KISCON\  NormalizeCompanyName.xaml | 회사이름에 주식회사가 있으면 (주)로 변경해서 처리  Input  - in\_ColumnIndex: int32 - 상호가 있는 Column 인덱스 정보  - in\_Type: String - KISCON에 있는 정보인지, 아닌지 구분 KISCON or other  Input/Output  - io\_SubcontractInfo: DataTable - 하수급업체 정보 |
| KISCON\  Open.xaml | KISCON 화면에 접속 후 로그인 하는 절차까지 수행함  Input  - in\_Url:String - 접속해야 할 URL 정보  - in\_Username:String - 접속 계정 정보  - in\_Password:SecureString - 접속 계정 패스워드 |
| KISCON\  OpenAndSelectProject.xaml | 공사개요를 업데이트 한다.  크게 3개의 계약체결 방식( 장기계속, 계속비, 일반)에 대해서 처리하며  장기계속의 경우 신규차수 생성 혹은 기존차수 변경에 대해서 처리한다.  정상 처리 이후 다음 단계로 넘어 간다.  Input  - in\_Config: Dictionary<String,Object> - 설정정보 맵  - in\_ConnStr: Dictionary<String, Object> - DB연결정보 맵  - in\_Query: Dictionary<String, Object> - 질의문 맵  - in\_TransactionItem : DataRow - 처리해야할 데이터  Input/Output  - io\_StatusMsg: Dictionary<String,String> - 진행상태 맵  Output  - out\_IsNewContract: Boolean - 신규차수 계약인지 여부 |
| KISCON\  SelectProject.xaml | 변경사항을 반영해야 하는 프로젝트를 선택하는 절차를 포함한다.  Input  - in\_ContractName: String - 공사(프로젝트) 이름  - in\_ContractSeq: String - 차수 계약 정보 (필요시)  Output  - 없음 |
| KISCON\  SelectSubcontractCompany.xaml | 하수급인 등록과정시 업체명을 검색하여 업체를 등록하는 절차를 수행 함.  업체명 검색 후 해당 업체명으로 검색되는 조건중에서 대표자가 동일한 건에 대해서 선택 후 사업자 등록증 확인 필요  Input  - in\_CompanyName: String - 업체명  - in\_CompanyRepName: String - 대표자 이름, 공동대표인 경우 컴마(,)로 구분  Input/Output  - io\_PageNumber: Integer - 페이지 번호, 선택한 항목의 사업자등록 번호가 같지 않은 경우  - io\_IndexPage: integer - 페이지 내부 index (zero-base), 다음 항목을 선택하기 위해서 필요  Output  - out\_Found: Boolean - 검색결과 존재 여부 |
| KISCON\  UpdateContractor.xaml | 수급업체 정보를 변경하는 절차를 포함한다.  공동사업인 경우 지분율에 맞게 금액을 계산해야 한다.  Input  - in\_UpdateDate: String - 수정일  - in\_ContractAmount: String - 도급계약 금액  - in\_UpdateDate:String - 변경 일자 , yyyy-MM-dd 포맷  - in\_IsNewContract: Boolean - 신규 차수 계약 여부  Output  - out\_ContractDT: DataTable - 수급업체 정보  - out\_StatusMsg: String - 진행상태 정보 |
| KISCON\  UpdateGuarantee.xaml | 보증서를 업데이트하는 철차를 수행 함.  보증서가 발급 된 경우를 다 찾아서 등록 한다.  처리 절차는 아래와 같다.  보증금예치방법 - 보증서 선택  보증권 예치업체당  보증종류 당  보증서를 조회한 후 선택 & 저장  Input  - in\_ContractName: String - 공사/프로젝트 명  - in\_GuaranteeValidPerido: String - 보증서 유효 기간 (현재일 기준 몇달 전인지)  Output  - out\_StatusMsg: String - 보증성 등록 여부 |
| KISCON\  UpdateProjectOverview.xaml | 공사개요를 갱신하는 절차를 포함한다.  공사개요에 필요한 계약년월일, 착공년월일, 준공(예정)년월일,도급금액이 포함된다.  공사종류(장기계속, 계속비, 일반)에 따라 입력하는 값이 달라진다.  Input  - in\_ContractName: String - 공사명 ( 회차를 포함하지 않음)  - in\_ContractSeq: String - 금번 차수  - in\_AgreedDate: String - 계약년월일 yyyy-MM-dd 포맷  - in\_StartDate: String - 착공년월일, yyyy-MM-dd 포맷  - in\_CompleteDate: String - 준공(예정) 년월일 yyyy-MM-dd 포맷  - in\_ContractAmount: String - 도급금액  - in\_FirstStartDate: String - 총공사 작공년월일, yyyy-MM-dd 포맷  - in\_FirstCompleteDate: String - 총공사 준공년월일 yyyy-MM-dd 포맷  - in\_FirstTotalAmount: String - 총공사부기금  - in\_UpdateDate: String - 변경 일자 yyyy-MM-dd 포맷  - in\_ContractType: String - 공사 구분( 계속비,장기계속,일반) 3가지로 구분  - in\_NewContractSeq: Boolean - 신규차수 여부  Output  - out\_StatusMsg: String - 공사개요 변경 정보 |
| KISCON\  UpdateReceiveMoney.xaml | 공사대급 수금 절차를 수행함.  수급인탭에서 공동사업인 경우 참여 사업자들의 지분율을 알아야 계산할수 있음  DooIT에서 넘겨주는 금액은 고객사(주)의 수령 금액이며, 이 금액을 기준으로 전체 금액을 계산하고, 각 회사의 지분율을 고려해서 각 회사의 수령금액을 입력해야 함  Input:  - in\_ReceiveInfo: DataTable - 공사대금 수금 내역 table  - in\_CompanyDT: DataTable - 수급업체 정보 (이름 및 지분율 포함)  - in\_BaseRefDate:String - KISCON 적용 기준 일자 ( 이 일자 이후 것만 처리 함)  - in\_ConnStrDict: Dictionary<String,Object> - DB연결정보 맵  - in\_QueryDict: Dictionary<String,Object> - 쿼리 맵  - in\_ConfigDict: Dictionary<String,Object> - 설정 맵  - in\_ContractName: String - 공사이름  Output  - out\_StatusMsg1: String - 기성 처리 내용  - out\_StatusMsg2: String - 선급금 처리 내용 |
| KISCON\  UpdateSubcontractGuarantee.xaml | 하수급인 보증서 업데이트 철차  Input  - in\_SubcontractInfo: DataTable - 하수급업체별 정보  - in\_ContractName : String - 공사(계약) 명  Input/Output  - io\_GntIssuers: List<String> - 보증서발급 업체 리스트  - io\_Companines: List<String> - 하도급업체 리스트  Output  - out\_StatusMsg: String - 보증서 등록 여부 |
| KISCON\  UpdateSubcontractInfo.xaml | 하수급인 등록 절차를 처리함.  하수급업체 리스트에서 계약회차기 0이면 신규 업체 등록이 필요하며, 아니면 계약 변경(기간,금액)만 필요 함  Input  - in\_SubcontractInfo: DataTable - 하수급업체 리스트 정보  - in\_ShareRatio: Dobule - 지분율 (단독, 공동 구분)  - in\_ContractName: String - 공사(계약)명  - in\_BaseRefDate:String - KISCON 적용 기준 일자 ( 이 일자 이후 것만 처리함)  - in\_ConnStrDict: Dictionary<String,Object> - DB연결정보 맵  - in\_QueryDict: Dictionary<String,Object> - 쿼리 맵  - in\_ConfigDict: Dictionary<String,Object> - 설정 맵  - in\_ContractName: String - 공사이름  Input/Output  - io\_ErrorList : List<string> - 오류 메세지를 저장하는 리스트  Output  - out\_StatusMsg1: String - 신규 등록 여부  - out\_StatusMsg2: String - 변경 등록 여부  - out\_SubStatusMsg: String - 하수급업체 보증서 등록 여부 |
| KISCON\  UpdateSubcontractPayment.xaml | 하도급대금 지급 내역을 갱신한다.  Input  - in\_Payments: DataTable - 하도금대급 지급 내역  - in\_Contracts: DataTable - 하수급인업체 정보 (이전단계에서 공종을 업데이트 함)  - in\_BaseRefDate:String - KISCON 적용 기준 일자 ( 이 일자 이후 것만 처리 함)  - in\_ConnStrDict: Dictionary<String,Object> - DB연결정보 맵  - in\_QueryDict: Dictionary<String,Object> - 쿼리 맵  - in\_ConfigDict: Dictionary<String,Object> - 설정 맵  - in\_ContractName: String - 공사이름    Input/Output  - io\_ErrorList : List<String> - 오류 내용을 저장하는 리스트  Output  - out\_StatusMsg1: String - 기성 지급  - out\_StatusMsg2: String - 선급금 지급 |

## 운영 환경 설정

개발 및 운영에 아래의 버전을 사용하기로 한다.

|  |  |
| --- | --- |
| UiPath RPA 콤포넌트 | 버전 |
| 오케스트레이터 | 2018.4.1 (LTS) |
| 로봇 | 2018.4.5 (LTS) |
| 스튜디오 | 2018.4.5 (LTS) |

### 로봇 PC 설정

운영에는 unattended 로봇을 사용하고, 로봇을 구동할 PC는 아래와 같이 설정하여야 한다.

|  |  |
| --- | --- |
| 항목 | 설정 |
| OS | 한글 Windows 10 Enterprise |
| 윈도우 업데이트 | Windows Update 서비스를 종료하고 “사용 안 함”으로 변경해야 함. |
| 화면 해상도 | Full HD (1920x1080) |
| 도메인 | 고객사 도메인 (필요시) |
| 웹브라우저 | Internet Explorer 11  "인터넷 옵션 -> 개인정보 -> 팝업 차단 사용" 체크해야 하고, 허용된 사이트도 없어야 함. |
| 오피스 | Microsoft Office 2010+ |
| Oracle | Oracle Client 11g 32-bit  아래와 같이 Windows registry 변경 필요:  "HKEY\_LOCAL\_MACHINE\SOFTWARE\WOW6432Node\ORACLE\KEY\_OraClient11g\_home1"에서 "NLS\_LANG"의 값을 "AMERICAN\_AMERICA.UTF8"으로 수정. |
| KISCON Plugin Control | [www.kiscon.net](http://www.kiscon.net) 사이트 접속시 필요한 컨트롤로 설치가 되어야 함 |

### 오케스트레이터 테넌트

오케스트레이터 관리자 페이지는 web interface로 접근할 수 있고 URL은 [{고객사](https://dscrpa.doosan.com/) Orchestrator URL} 이다. 오케스트레이터에 생성되어 있는 테넌트는 아래와 같다.

|  |  |  |
| --- | --- | --- |
| Tenant 이름 | 관리자 ID | 용도 |
| HOST | admin | 테넌트와 라이센스 관리 |
| Default | admin | 오케스트레이터 설치 시에 기본으로 생성되는 테넌트.  사용하지 않음. |
| DEV | admin | 개발용 테넌트 |
| TEST | admin | 테스트용 테넌트 |
| PROD | admin | 운영용 테넌트 |

admin 계정을 사용하지 말고 관리자마다 별도의 ID를 부여하여 사용하여야 한다. 하나의 ID로 한 번에 하나의 세션만 허용하기 때문에, 동일한 ID로 로그인하면 기존의 세션은 끊어지게 된다.

### 로봇 그룹

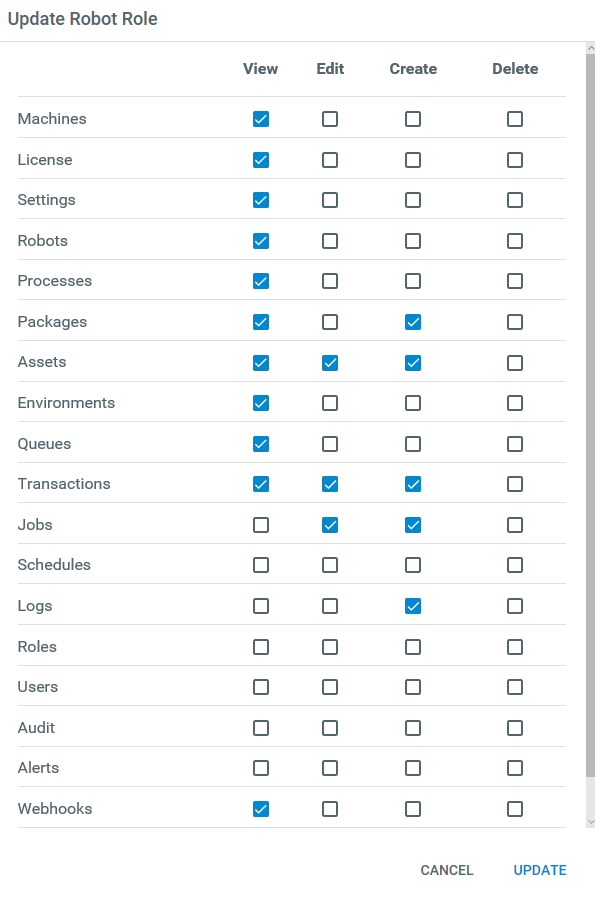
오케스트레이터에 고객사 환경에 맞는 이름으로 Environment를 생성하고 KISCON 업무를 수행할 로봇들을 모두 배속시켜야 한다.

### 로봇 권한

로봇이 프로세스를 invoke할 수 있게 하려면 오케스트레이터에 관리자 권한으로 로그인하여 “Robot”의 Role에 추가로 권한을 부여해야 한다. Processes와 Environment에 대하여 View 권한을, 그리고 Jobs에 대하여 Edit과 Create 권한을 부여해야 한다.

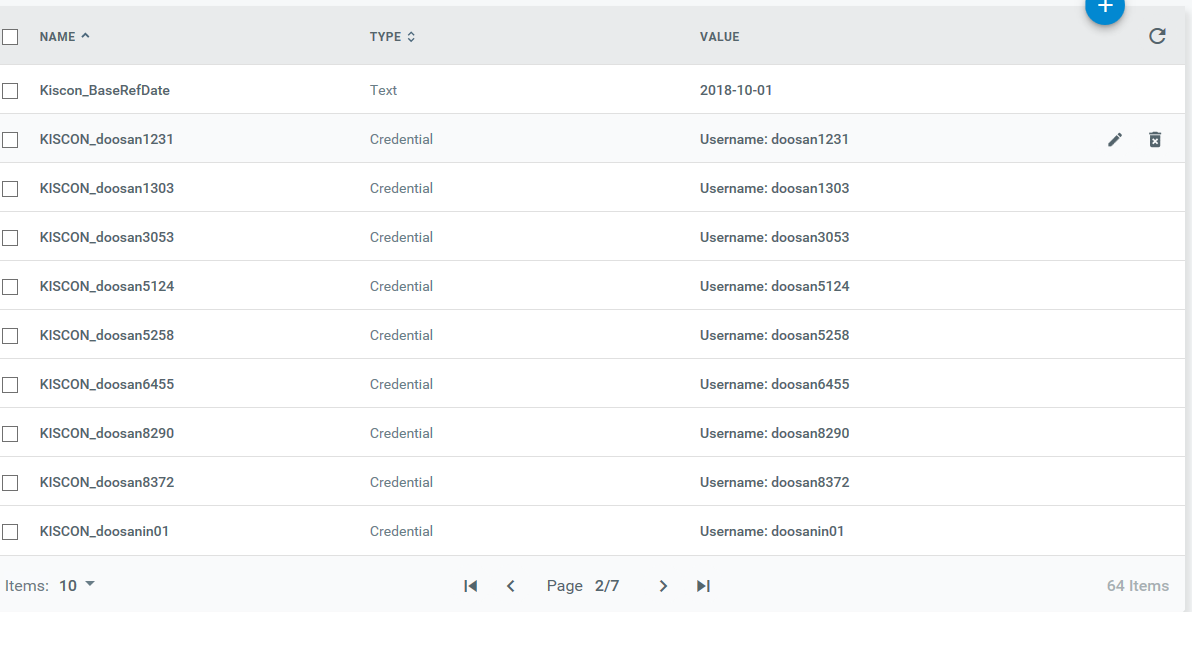
또한, 로봇이 오케스트레이터 asset에 값을 설정하고 신규로asset을 만들 수 있게 하려면 Edit 과 Create 권한을 줘야 한다.

따라서 로봇의 권한 설정은 아래와 같아야 한다.



### 오케스트레이터 asset

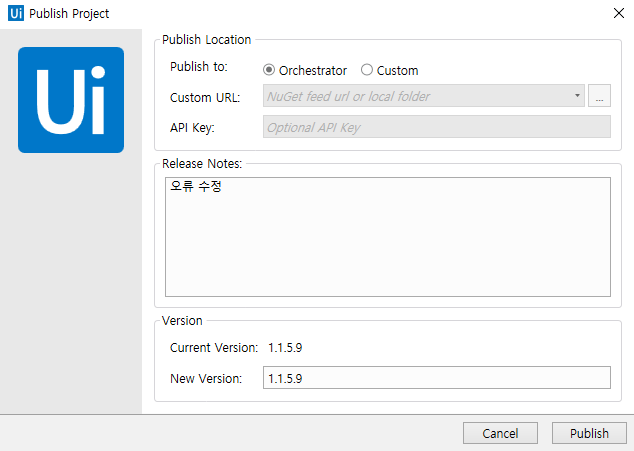
KISCON 프로세스를 구동하기 위하여 필요한 오케스트레이터 asset은 아래와 같다.



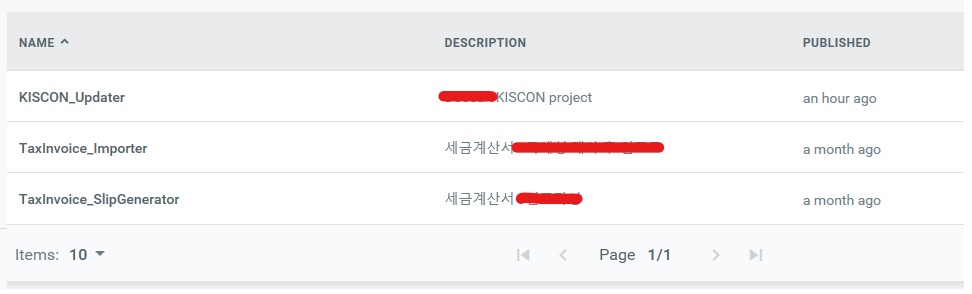
### 패키지 등록

프로세스가 수정되면 아래와 같은 순서로 새 버전의 패키지를 오케스트레이터에 등록해야 한다.

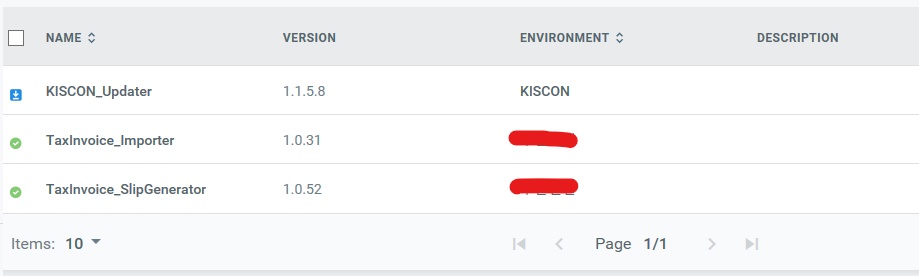
1. UiPath Studio에서 새 버전 번호를 부여하여publish함



1. 오케스트레이터에 관리자 권한으로 로그인하여 패키지를 업로드함, Publish 위치를 Orchestrator로 지정하게 되면 자동으로 등록 됨



1. 새 버전의 패키지를 사용하도록 지정함.



## 개발자 가이드

### 프로세스 명명 규칙

- 영어만 사용

- 각 단어의 첫 문자는 대문자, 나머지 문자는 모두 소문자

- KISCON 프로세스는 접두어 "KISCON\_"로 시작

- 명사형을 사용

### 프로젝트 폴더 구조

- Data: 데이터 및 설정 파일들

- Framework: 프레임워크 xaml 파일들

- Custom: 고객사 내부 데이터를 다루는 xaml 파일들

- KISCON: KISCON사이트를 다루는 xaml 파일들

### XAML 파일 명명 규칙

- 영어만 사용

- 각 단어의 첫 문자는 대문자, 나머지 문자는 모두 소문자

- 동사+목적어 형태로 명명함

바른예) ComposeUrl

틀린예) UrlComposer

### XAML 파일 테스트 방법

- 해당 XAML 파일의 variable/argument의 default value에 테스트 데이터를 설정하여 테스트하면 안됨

- 테스트용 stub을 별도로 생성하여 해당 XAML 파일을 호출하는 방식으로 테스트해야 함

- 테스트용 stub 파일명은 접두어 "Test\_"를 앞에 붙이는 방식으로 명명함

### Annotation 규칙

- 한글 사용 가능

- 모든 XAML 파일의 최상위 컨테이너는 반드시 annotation을 가져야 하고 아래의 내용을 포함해야 함

Pre-condition

Input

Output

Outcome

### variable 명명 규칙

- 영어만 사용

- 각 단어의 첫 문자는 대문자, 나머지 문자는 모두 소문자

바른예) ComposeUrl

틀린예) ComposeURL

### argument 명명 규칙

- 영어만 사용

- 각 단어의 첫 문자는 대문자, 나머지 문자는 모두 소문자

- IN argument는 접두어 "in\_"로 시작

- OUT argument는 접두어 "out\_"로 시작

- INOUT argument는 접두어 "io\_"로 시작

### activity 명명 규칙

- 기본 activity 이름은 그대로 유지

- 기본 activity 이름 뒤에 " - "를 붙이고 간단한 description을 추가

- 간단한 description으로 설명이 부족한 경우 annotation에 상세 description 추가

### package version numbering 규칙

- major.minor.build 형태로 사용

- UiPath Studio가 자동으로 생성하여 제시하는 숫자는 사용하지 않음

- major 번호는 1부터 시작하고 minor와 build 번호는 0부터 시작. 따라서 첫번째 버전 번호는 1.0.0

## 운영 환경 전환

개발환경에서 운영환경으로 전환하려면 아래와 같은 변경이 필요하다.

|  |  |
| --- | --- |
| 항목 | 변경할 내용 |
| 고객사 DB | 고객사 DB에 대한 쿼리와 접속 정보를 운영환경에 맞게 바꾸어야 한다. |
| KISCON 사용자정보 | KISCON 사용자 아이디 및 비밀번호는 자동으로 운영환경에 등록하기 위해서는 패스워드 부분이 공백이 되면 안된다. 등록 이후 공백으로 자동 변경되며, 수정이 필요한 경우 고객사DB에서 수정해줘야 한다.  (AddCredentialAssts.xaml 파일에서 주석된 부분 제거 필요) |
| 탭별 타임아웃정보 변경 | Config.xlxs에 설정된 각 탭별 timeout 시간(분단위)를 조정할 수 있다. |
| 대상프로젝트 조회 | 프로젝트 조회 조건은 최근 변경일, 프로젝트 코드 낮은 순서, 변경차수 낮은 순서, 변경회수 낮은 회수 순서임, 순서 변경서 Config.xlsx Queries 탭에서 GetProjectOverview 질의문을 수정하면 된다.  작업중인 P 상태에서 N으로 변경하는 코드는 Custom \ UpdateInProgressStatus.xaml 이다. |
| Studio 및 Robot 버전 업그레이드 | 아래 링크에서 최신 버전을 다운로드 받아 설치한다. 설치 파일로 설치시 업그레이드 된다.  Studio & Robot : <https://download.uipath.com/UiPathStudio.msi>  설치시 Robot만 설치하는 경우 Advanced 옵션에서 Robot 부분만 체크하고 Studio 부분은 체크를 해제해주면 된다. |