Assessments - Perform an OPV and OEC

Last Modified on 23/01/2023 1:20 pm ACDT

Patient assessments unlock powerful capabilities to verify your patient, or their eligibility with their health fund.

Initial setup

Create an assessment.

Add the following assessment elements:

- Patient ID
- Given Name
- Family Name
- Sex
- Date of Birth
- Gender
- Location
- Fund details
- Medicare OPV
- Medicare OEC

Required fields marked with *

Given Name*	Luke
Middle name	
Family name*	Alford
DOB*	11/03/1950 🗓
Gender*	Male ✕ ▾
Location*	Adelaide ✕ ▾
Last name alias	
First name alias	
	OPV

Fund Details

Fund	Fund Number	Fund IRN
Mock HF (TST) ✕ ▾	123456	1

Online Eligibility Claim Type	Fund ▾
Admission Date	04/11/2022 🗓
Length Of Stay	0
Presenting Illness Code	Assisted Reproductive services or infertility treatments ✕ ▾
Presenting Illness MBS Number	Select an MBS item number ▾
Additional Services	Item Cost Add Service

☐ Pre-existing Condition

☐ Compensation Claim

Performing the Assessment

When these fields are available, the OEC element will allow requests to Medicare/the health fund.

Advanced usage: Trigger Scripts to create an Estimate

Create a new trigger class; and within that, a new trigger script:

Trigger Class Details

Type	Assessment
Name	Estimate and OEC
Version	1.0

Columns

Search... 🔍

Name	Datatype

Trigger Scripts

Search... 🔍

New Script

Name	Version	Trigger	State	Active	
Create Estimate, OEC and Send It Places	1.0	Manual Script	U	Yes	edit remove deactivate

Library Code

Library Code

Add behaviour similar to the below; using the **account.create_estimate** method.

Code*

```
1  console.log("1. Check required information is present")
2
3
4  console.log("2. Create the estimate")
5
6  var patient = assessment.patient
7  var staff_member = assessment.e229234540_staff_picker
8  var guarantor_id = assessment.fund
9  var location = finder.location.by_name('ADL') // hardcoded location. TODO: extra code 2 looks like a location
10 var account = patient.find_or_create_patient_account(location, guarantor_id, staff_member)
11 var service_location = location.inpatient_service_locations[0] // first IP service location
12
13 var items = new Array
14 if (assessment.item1) items.push({ code: assessment.item1 })
15 if (assessment.item2) items.push({ code: assessment.item2 })
16 var options = { referral: 'none', note: 'This is a test', mpr: true, extra_code_1: assessment.extra_code_1, extra_code_2: assessment.extra_code_2
17
18 // TODO: Do ... something? with the OEC's copay/excess
19
20 var estimate = account.create_estimate(staff_member, service_location, items, options)
21 if (!estimate) { script.fatal("Error creating estimate") }
22 alert('Created estimate number ' + estimate.id + ': https://test3.use.careright.com.au/estimates/' + estimate.id)
23
24 console.log("Estimate successfully created", estimate);
25
26 console.log("3. Print... PDFs")
27 /*
28 assessment.print('My Template', {
29   patient: { given_name: 'Mary' },
30   assessment: { my_element: 'New Value' },
31   my_namespace: { days_since_last_accident: 10 }
32 })
33
34 // Resulting in the following fields being supported in the PDF template:
35 // * patient[given_name]
36 // * assessment[my_element]
37 // * my_namespace[days_since_last_accident]
38
39
40 */
```

Publish and associate your trigger script.

Which triggerable classes does this assessment support?

☐ Admission Form 8

☐ Button Trigger 1.0

☑ Estimate and OEC

Advanced usage: Trigger Scripts to Start a Pre Admission

To make an extra control available when viewing the assessment, a trigger script of the type *Approved Manual Script* can be created.

| Pre Admit | 1.0 | Approved Manual Script | U | Yes | Edit | Remove |
| | | | | | Deactivate | |

Here, we simply map a number of pre-filled details from the OEC element and response, into URL parameters.

The pre-admission will be created with the details set.

Code

```
 1  /* Example assessment structure
 2  {
 3    {"id"=>2,
 4    "assessment_id"=>14,
 5    "created_at"=>2023-01-19 02:46:46 UTC,
 6    "updated_at"=>2023-01-19 02:53:23 UTC,
 7    "completed_at"=>2023-01-19 02:53:21 UTC,
 8    "run_by"=>nil, "patient_id"=>"17685", "prn"=>"18596015", "date"=>"", "estimate_id"=>46,
 9    "min_estimate_id"=>nil, "given_name"=>"Test", "middle_name"=>"Example",
10    "family_name"=>"Careright", "date_of_birth"=>Mon, 11 Jan 1999, "gender"=>"2",
11    "crn"=>"PN1859601", "location"=>"2", "last_name_alias"=>"Angel", "first_name_alias"=>"",
12    "med_number"=>"", "medically_necessary"=>true,
13    "oec_details"=>"{\"medicare_online_request_id\":38,\"additional_services\":[{\"item_number\":\"13203\",\"cost\":\"102\"},{\"item_number\":\"13212\",\"cos
14  */
15
16
17  parts = [];
18  if (assessment.medically_necessary && assessment.oec_details) {
19    oec = JSON.parse(assessment.oec_details)
20
21    if (oec['excess_amount']) {
22      parts.push("admission[excess]=" + oec['excess_amount'] / 100)
23    }
24    if (oec['copayment_amount']) {
25      parts.push("admission[co_payment]=" + oec["copayment_amount"] / 100)
26    }
27    parts.push("admission[funding_choice]=Health%20Fund")
28    parts.push("admission[fund]=" + assessment.fund)
29    parts.push("admission[fund_number]=" + assessment.fund_number)
30    parts.push("admission[fund_irn]=" + assessment.fund_irn)
31    parts.push("admission[pres_srv]=Assisted%20Reproductive%20Services")
32  } else {
33    parts.push("admission[funding_choice]=Other")
34  }
35
36  redirect("/patients/" + assessment.patient_id + "/admissions/new?state=PreAdmit&" + parts.join("&") )
37
```