BG3

Forfatter:
BRIAN/HOUMANN

VEJLEDNING TIL IMPLEMENTERING AF TRACKING FOR TEKNIKLEVERANDØRER AF PORTALSERVICER PÅ BORGER.DK (TÆLLERSCRIPT)

Version 2.7

Indholdsfortegnelse

1. Inden start!

1.1 Vigtig opdatering!

I vejledningen vejledes der til implementering af både "start-script" og "slut-script". Det anbefales stadig at benytte begge scripts, hvis det er muligt, men det er gjort frivilligt. Det er derfor nu muligt blot at implementere "slut-script".

1.2 Hvorfor tællerscript?

Installationen af tællerscript i digitale selvbetjeningsløsninger gør det muligt at automatisere indsamlingen af vigtig forretnings- og ledelsesinformation – dvs. antal af startede og afsluttede transaktioner.

Borger.dk's tællerscript er derfor ikke en erstatning for leverandørerne eksisterende statistikværktøj, men er en vital hjælp, der giver løsningsejere et samlet overblik af deres portefølje af digitale selvbetjeningsløsninger – dvs. definition af påbegyndte (en "start") og succesfuldt afsluttede (et "slut") transaktioner på den digitale kanal – på tværs af løsninger og serviceområder.

Tællerscript er gratis at bruge og kræver kun implementering. Installationen af tællerscript er et krav i udviklingsvejledningen for velfungerende selvbetjeningsløsninger[1] samt et krav som borger.dk stiller til alle borgerrettede løsninger – dvs. tællerscript skal installeres.

1.3 Beskrivelse af løsningen

Der ønskes en central tracking af portalservices tilknyttet borger.dk med opsamling af data på påbegyndte borger-aktiviteter og afsluttede borger-aktiviteter.

Husk, en borger-aktivitet og portalservice er defineret som en servicetransaktion, fx en ansøgning om pension, ansøgning om tilskud, en anmeldelse af rotter osv. Definitionen betyder også at hjælpe funktioner i en løsning, fx en regnemaskine funktion, ikke skal ses som en servicetransaktion.

- En påbegyndte borger-aktivitet (en "start") skal fortolkes som starten på en digital selvbetjening, dvs. når en bruger klikker på fx start-knappen). En selvbetjeningsløsning kan rumme flere forskellige indberetningsforløb, som det er vigtigt, at løsningsejeren får særskilt statistik på, fx opskrivning til daginstitution og ansøgning om økonomisk friplads. Det betyder, at der ofte skal placeres flere start- og stopscripts indenfor selvsamme løsning.
- En afsluttede borger-aktivitet (et "slut") skal fortolkes som gennemført og afsluttet en digital selvbetjening, dvs. når en bruger klikker på fx indsend- eller afslut-knappen.

Løsningen er baseret på en central tracking-server, der modtager data gennem et JavaScript, som placeres på de enkelte portalservicer.

Der placeres to separate JavaScript-blokke på hhv. den side, hvor borgeren påbegynder en aktivitet – fx siden med første trin af konkret selvbetjeningsløsning – og den side, der afslutter borgerens aktivitet – fx kvitteringsside for gennemført selvbetjening. Dette kan gøres på flere måder afhængig af arkitekturen på den enkelte portalservice.

[1] Udviklingsvejledning for velfungerende selvbetjening http://arkitekturguiden.digitaliser.dk/godselvbetjening

Husk at et af målene med tællerscriptet er at sikrer en ensartet registrering af borgernes initieringer og gennemførte transaktioner for selvbetjeningsløsninger. Her følger en gennemgang af, hvordan henholdsvis start- og stopscript bør placeres for forskellige typer af indberetningsforløb.

OBS. Der er nu kun krav om slut-script, start-script er valgfrit, men fortsat anbefalet.

Løsningstype	Eksempel på opgave/løsning med denne karakteristik	Use case	Typiske tvilvsspørgsmål	Sådan placeres tællerscript
Løsning med flere parter – lineær	Vielse	Part 1 logger ind, udfylder og inviterer part 2. Part 2 udfylder yderligere oplysninger og signerer/afslutter forløb.	Er der tale om en eller flere initieringer? Er der tale om en eller flere transaktioner?	Der er her tale om et generelt eller simpelt flow med en initiering, når 1. part tilgår løsningen og herefter sendes til 2. part som afslutter og indsender ansøgningen. Dvs. at det endelige stopscript placeres når den samlede transaktion indsendes til myndigheden. Flowet bliver: /start (dvs. start for 1. part) /start -> /slut (dvs. start og slut for 2. part)
Løsning med flere parter – tilbageløb	Lokale og-aktivitetstilskud (foreninger)	Forening ansøger om tilskud og signerer/afslutter. Kommune modtager, redigerer i oplysningerne og sender tilbage til foreningen, da noget ikke var udfyldt korrekt. Forening berigtiger/retter oplysninger og sender tilbage. (osv)	Er der tale om en eller flere initieringer? Er der tale om en eller flere transaktioner?	Der er her tale om et komplekst flow. Hver gang foreningen (parten) udfylde og indsender en ny ansøgning eller tilretter en ansøgning og indsender, er der tale om en initiering, hvor der skal placeres et start- og stopscript. Det skal dog være muligt ud fra de metadata, som er tilknyttet tællerscriptet, være muligt at identificere, om der er tale om oprettelse af en ny sag eller ændring af status på en tidligere indsendt ansøgning. Flowet kan være: /start -> /slut/indsend /start -> /slut/opdateret
Løsning med 1 part – lineær	Anmeldelse af rotter	Borger anmelder og indsender.		Der er her tale om et generelt eller simpelt flow. Startscript placeres på den side, hvor indberetningen starter. Stopscript afvikles, når borgeren klikker på "indsend" (eller tilsvarende).
Løsning med flere parter på samme ansøgning – lineær	Navne og adressebeskyttelse (Bestilling af sundhedskort)	Borger ansøger som navne og adressebeskyttelse og tilvælger automatisk alle hjemmeboende børn under 18 som en af den samme digitale indberetning.	Er der tale om en eller flere transaktioner, når nu den ene transaktion gælder for flere personer? I den gamle papirverden, kunne der potentielt set jo sagtens have skulle været foretaget flere transaktioner/udfyldt 1 blanket pr. person.	Alt efter løsningens kompleksitet kan der her være tale om et generelt, simpelt eller komplekst flow. Der er dog kun tale om én transaktion. Dog er det vigtigt forretnings- og ledelsesinformation at få det fulde antal bestillinger, fx antallet af bestilte og udstedte sundhedskort, i indberetningsforløbet med. Derfor skal start- og stopscriptet placeres således at for hver afviklet transaktion også kommer et parameter med for antallet af bestillinger.

				Flowet kan være: /start -> /slut/antallet af bestillinger (fx dynamisk numerisk parameter for antallet af bestilte sundhedskort).
Løsning med andre funktioner	Boligstøtte	Borger foretager vejledende beregning, men indsender ikke en ansøgning	Skal der kun tælles, når noget indsendes?	Alt efter løsningens kompleksitet kan der her være tale om et generelt, simpelt eller komplekst flow. Der er dog ikke tale om en transaktion, da der ikke oprettes en ny sag eller ændres på en eksisterende. Der kan dog med fordel placeres et start- og stop script på løsningen, da det giver et overblik over, hvor mange borgere der modtager information om ydelsen. Såfremt tællerscriptet bliver anvendt, skal det tydeligt fremgå at metadata, at der er tale om informationer og ikke transaktioner.
Løsning med "gem til genoptag"	Ansøgning om byggetilladelse	Borger går i gang med at udfylde, men gemmer de indtastede oplysninger og afbryder forløb. Borger optager forløbet på et senere tidspunkt.	Er der tale om en eller flere initieringer?	Der er her tale om et komplekst flow, da der er tale om flere initieringerne. Startscriptet skal placeres, hver gang en borger tilgår løsning på ny, og stopscriptet når ansøgningen indsendes. Startscriptet vil således blive aktiveret flere gange uden at blive afsluttet.
Løsning med kommuneinitieret forløb – tilbageløb	Samtykkeerklæring	Kommune udfylder oplysninger og inviterer borger til at fortsætte forløbet. Borger videreudfylder og returnerer til kommune.	Tæller kommunens handlinger som initiering?	Der er her tale om et generelt eller simpelt flow, da kommunens handlinger ikke tæller som initiering. Startscriptet placeres hvor borgeren tilgår løsning på baggrund af invitation fra kommunen for at videreudfylde. Stopscriptet placeres, hvor borgeren afsluttet og indsender erklæringen.
Løsning med straks-afklarende spørgsmål	Ansøgning om udvidet helbredstillæg	Borgeren vil ansøge, men når ikke at logge ind, da et straks-afklarende spørgsmål fortæller, at borgeren skal være pensionist for overhovedet at kunne søge.	Er der tale om initierering? Er der tale om transaktion?	Der er her tale om et generelt eller simpelt flow, da der ikke er tale om en initering, før borgeren begynder det faktiske indberetningsforløb. De indledende spørgsmål er ikke en del af transaktionen.

2. Hurtig vejledning

OBS. Der er nu kun krav om slut-script, start-script er valgfrit, men fortsat anbefalet.

Følgende kode indsættes som HTML på startsiden til servicen:

```
<script type="text/javascript">
document.write(unescape("%3Cscript src='" + (("https:" == document.location.protocol) ? "https://" :
"http://")+ "tracking.danmark.dk/urchin.js' type='text/javascript'%3E%3C/script%3E"));
</script>
<script type="text/javascript">try {urchinTracker("/PORTALSERVICEID/start");} catch(err) {}</script>
```

Følgende kode indsættes som HTML på siden, hvor borgeren har afsluttet servicen:

```
<script type="text/javascript">
document.write(unescape("%3Cscript src='" + (("https:" == document.location.protocol) ? "https://" :
"http://")+ "tracking.danmark.dk/urchin.js' type='text/javascript'%3E%3C/script%3E"));
</script>
<script type="text/javascript">try {urchinTracker("/PORTALSERVICEID/slut");} catch(err) {}</script>
```

Koden indeholder elementet "PORTALSERVICEID". Her skal leverandøren indsætte egne oplysninger til identifikation af den relevante service. Dette uddybes nedenfor. Kode-stumperne skal placeres sidst på siden umiddelbart foran tagget </body>.

PORTALSERVICEID skal erstattes med en af følgende to muligheder:

1. Tæller-id

En unik kode (PORTALSERVICEID, der i statistiksystemet kaldes "Tæller-id"), en såkaldt GUID der leveres af admin@borger.dk.

Denne GUID er unik for enhver kombination af en selvbetjeningsløsning og en abonnent. En løsning som er udbredt til 80 kommuner vil således have 80 forskellige GUID'er tilknyttet i statistiksystemet, som skal anvendes for at opnå statistik for hver abonnent.

2. Alternative/egne ID'er

Hvis ikke der bruges "Tæller-id" kan leverandørens egne ID'er benyttes indsat I en streng med følgende format:

```
/SupplierFormat/SupplierID/SuppliersTechnicalSolutionID/MunicipalityCode
```

Leverandørens eget ID indrapporteres til admin@borger.dk.

HTML-koden som skal indsættes på siden kan eksempelvis se således ud:

```
<script type="text/javascript">
document.write(unescape("%3Cscript src='" + (("https:" == document.location.protocol) ? "https://" :
"http://")+ "tracking.danmark.dk/urchin.js' type='text/javascript'%3E%3C/script%3E"));
</script>
<script type="text/javascript">try
{urchinTracker("/SupplierFormat/SupplierID/SuppliersTechnicalSolutionID/MunicipalityCode/slut");}
catch(err) {}</script>
```

De enkelte elementer i strengen skal erstattes med data som identificerer løsningen. Nedenstående tabel forklarer termer, giver eksempler og indikerer hvor man kan få et eventuelt ID eller kode til JavaScript-blokkene.

Term	Forklaring, eksempel og hjælp
SupplierFormat	SupplierFormat angiver for statistiksystemet, at der kommer en streng med parametre. Dette led i strengen skal ikke ændres eller overskrives. Husk / før teksten "SupplierFormat". Og Husk store bogstaver de rigtige steder.
SupplierID	SupplierID er leverandørens ID i statistiksystemet. *Eksempel:* *Her udskiftes "SupplierID" med fx "8256".* Hvis SupplierFormat ikke kendes, kan dette fås ved at kontakte borger.dk (admin@borger.dk, skriv *"SupplierID"* i emne linjen).
SupplierTechnicalSolutionID	SupplierTechnicalSolutionID er et id – tekststreng eller talværdi – til servicen, som leverandøren bestemmer. Det anbefales at anvende et sigende ID. For at anvendelse af løsningen kan registreres korrekt af statistikløsningen, skal dette ID sendes til borger.dk. *Eksempel:* *Her udskiftes "SupplierTechnicalSolutionID" med fx "DigitalPladsanvisning".* Hvis et SupplierTechnicalSolutionID ikke er registreret i statistiksystemet sendes dette til borger. Borger.dk (admin@borger.dk, skriv *"registrering af SupplierTechnicalSolutionID"* i emne linjen).
MunicipalityCode	MunicipalityCode er den officielle kommunekode. Sendes ingen kode med opfattes servicen som at gælde for hele landet. *Eksempel:* *Her udskiftes "MunicipalityCode" med fx "762".* Hvis kommunekoden ikke kendes kan denne oplyses ved henvendelse til borger.dk (admin@borger.dk, skriv "oplysning af kommunekode" i emnefeltet).

Tabel 1 Forklaring, eksempler og hjælp til anskaffelse af ID og koder.

Den samlede streng der i dette tilfælde skal erstatte "PORTALSERVICEID" vil dermed se således ud (med fiktive data): `/SupplierFormat/8256/DigitalPladsanvisning/762/`

3. Uddybende vejledning

Herunder skitseres hvor JavaScript-blokkene skal implementeres i et generelt selvbetjeningsflow.

OBS. Vær opmærksom på, at det er blevet frivilligt at benytte både start- og slut-script. Slut-scriptet skal dog placeres.

3.1 Første del af JavaScript-blokken

Den første del af JavaScript-blokken henter et eksternt JavaScript ”urchin.js”, der ligger på serveren tracking.danmark.dk.

```
<script type="text/javascript">
document.write(unescape("%3Cscript src='" + (("https:" == document.location.protocol) ? "https://" :
"http://")+ "tracking.danmark.dk/urchin.js' type='text/javascript'%3E%3C/script%3E"));
</script>
```

Hvis den aktuelle portalservice kører på en krypteret protokol (https), vil det eksterne JavaScript ligeledes blive hentet på en krypteret protokol.

Denne blok er identisk for alle sider, der skal have tracking og skal inkluderes en enkelt gang på disse.
Der bliver ikke udført tracking i denne blok alene men blot hentet ekstern kode, og der er derfor ingen konsekvens ved at lade den inkludere på sider, der ikke skal trackes.

3.2 Anden del af JavaScript-blokken

Den anden del af koden vil lave et kald til JavaScript-koden, som udfører en tracking af sidevisningen til tracking-serveren. Kaldet ved start af en borgeraktivitet ser sådan ud:

```
<script type="text/javascript">try {urchinTracker("/PORTALSERVICEID/start");} catch(err) {}</script>
```

Strengen "/PORTALSERVICEID/start" er nøglen i differentiering af portalservicer og hhv. påbegyndelse og afslutning.

For den side, hvor borgeren påbegynder aktiviteten, skal strengen lyde "/PORTALSERVICEID/start”.
For den side, hvor borgeren afslutter aktiviteten, skal strengen lyde "/PORTALSERVICEID/slut”.
Leverandøren af portalservicen udskifter ”PORTALSERVICEID” med et tildelt ID for portalservicen.

3.3 Flere mulige afslutninger af en aktivitet

Hvis portalservicen har flere mulige afslutninger fordelt på flere sider, dvs. en forgrening fra det punkt, hvor aktiviteten starter, så kan der differentieres ml. disse slut-sider med en tekststreng eller et nummer.

Bemærk, at hvis løsningen lægger op til at brugerne kan afslutte flere gange indenfor samme session, skal nedenstående model ”Flere separate handlingsspor i en portalservice” bruges.

Koden til registrering af slutaktiviteten skal i dette tilfælde lyde "/PORTALSERVICEID/slut/TOKEN”, hvor *TOKEN* er en kort tekststreng, som kan give mere meningsfuld repræsentation på statistikløsningens front-end. Dette kan fx være angivelse af *ny* eller *opdateret*, hvis man vil angive om der er tale om en ny ansøgning eller opdatering til en allerede eksisterende sag.

Eksempler på anden del af JavaScript-koden til forskellige afslutninger ville være:

På den ene afslutningsside:

```
<script type="text/javascript">try {urchinTracker("/PORTALSERVICEID/slut/ny");} catch(err)
{}</script>
```

På den anden afslutningsside:

```
<script type="text/javascript">try {urchinTracker("/PORTALSERVICEID/slut/opdateret");} catch(err)
{}</script>
```

I alle tilfælde skal den første del af blokken med JavaScript-kode naturligvis også inkluderes.

3.4 Placering af koden

Kode-blokkene med JavaScript skal placeres sidst på siden umiddelbart foran tagget </body>.
Har man fordel af at placere det tidligere, fx i toppen af siden, kan dette også vælges. I så fald kan der dog ses en lille nedgang i sidens overordnede ydeevne.

Første del af kodeblokken skal altid placeres før anden del på en måde som sikrer, at der vil være en tidsforskydelse på minimum 1 sekund mellem afviklingen af de to blokke.

Kodeblokkende (scripts) kan implementeres på mange forskellige måder, afhængigt af hvordan den enkelte selvbetjeningsløsning er designet. Herunder kommenteres de forskellige metoder.

3.4.1 Generelt flow:

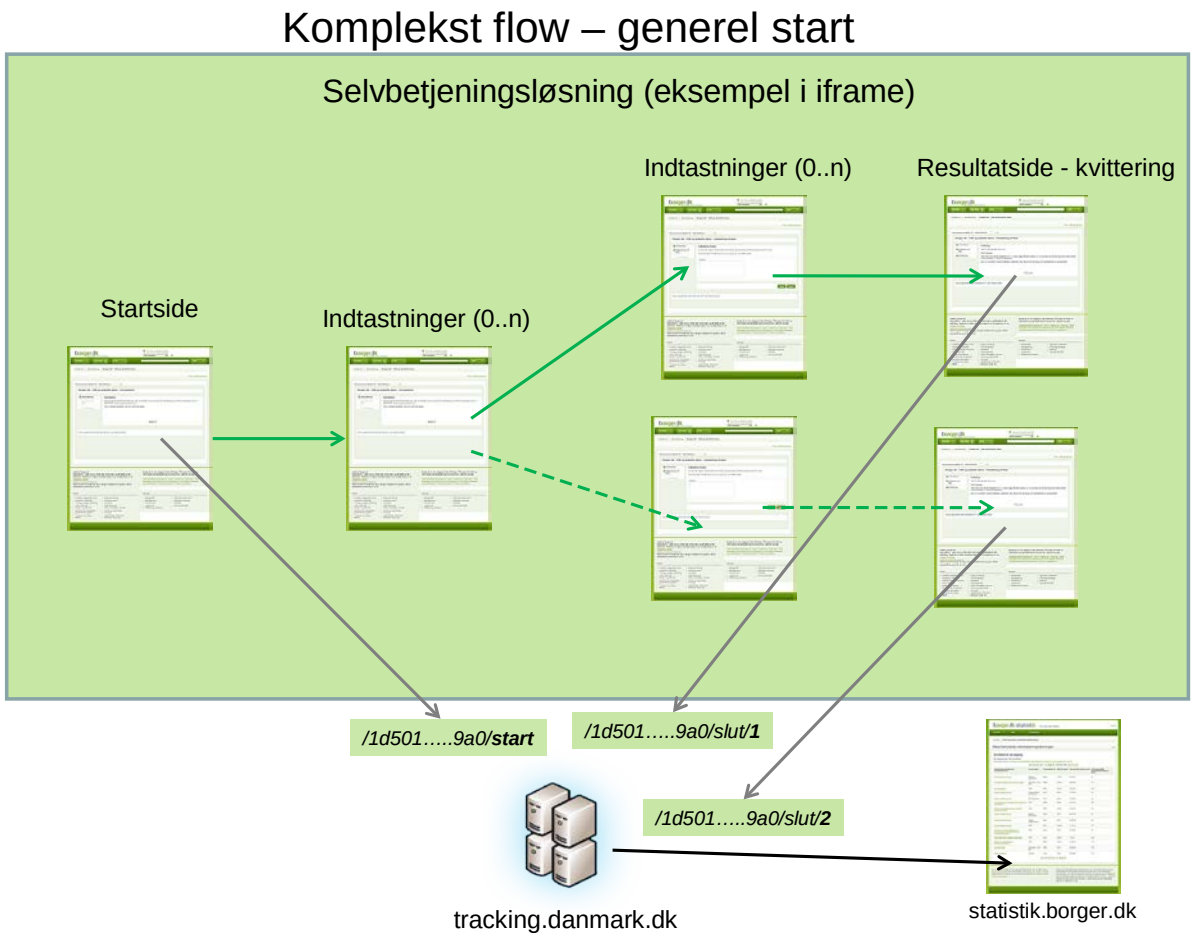
Startscript er placeret på den startside, hvor selve selvbetjeningsforløbet aktiveres af brugeren. Her tælles altså aktivering af selvbetjeningsløsningen, men ikke nødvendigvis konkret påbegyndt selvbetjeningsforløb. Slutscriptet er placeret på den side, hvor brugeren lander efter afsluttet selvbetjening.

3.4.2 Simpelt flow:

Startscript er placeret på den startside, hvor selve selvbetjeningsforløbet påbegyndes af brugeren. Fx første trin af x-antal trin. Her tælles konkret påbegyndt selvbetjeningsforløb, idet der tælles fra første trin af selve selvbetjeningsforløbet.

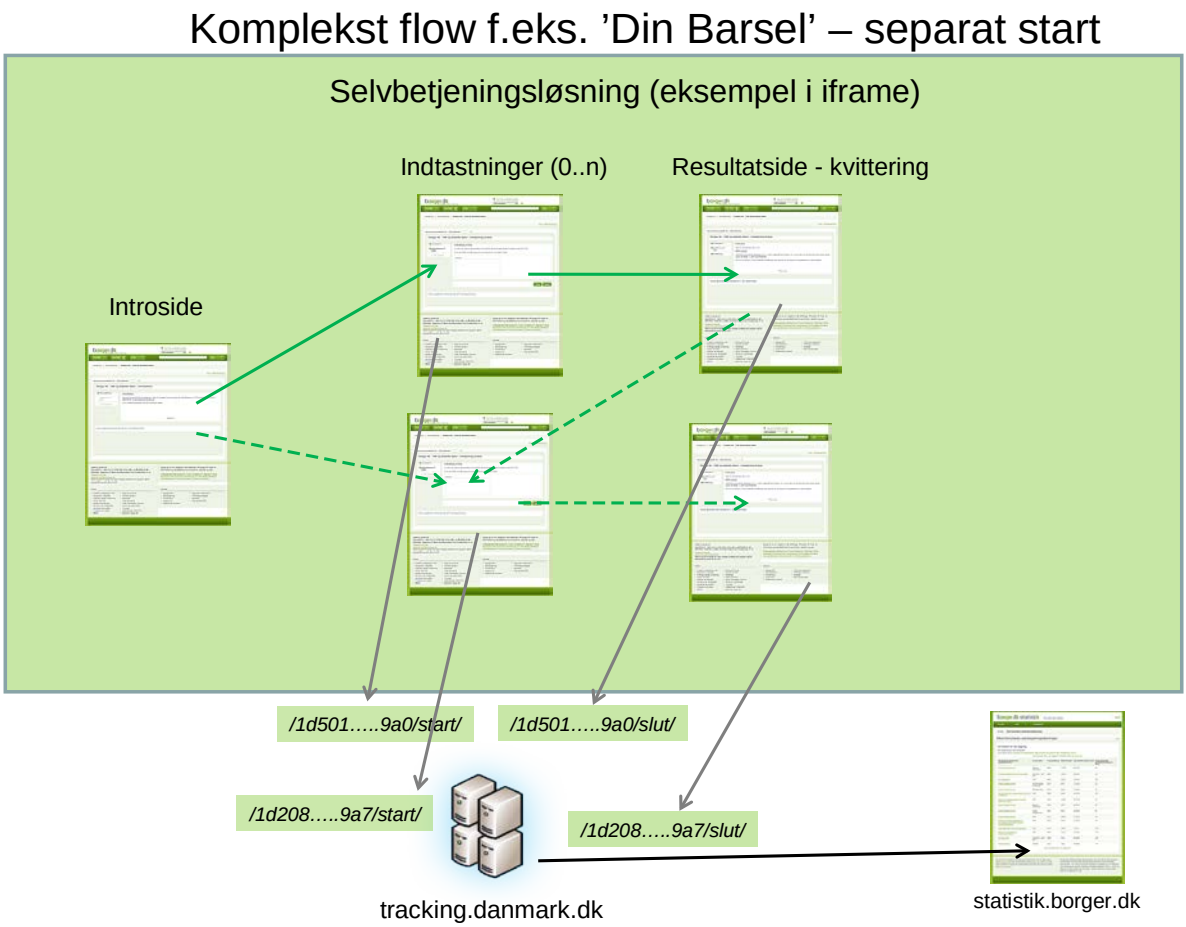
Komplekst flow – generel start:

Dette beskriver, hvordan tællerscripts skal implementeres på en løsning, som kan afsluttes på én af flere mulige måder. Startscriptet er placeret på første trin i løsningen. Der placeres samtidig navngivne slutscripts på sidste side af hvert af de mulige afslutningsspor. Denne implementering af tællerscripts er *ikke* egnet til løsninger, hvori brugeren kan gennemføre flere separate handlinger i én session, fx både beregne og ansøge. Den slags løsninger skal håndteres som nedenstående "Komplekst flow – separat start".



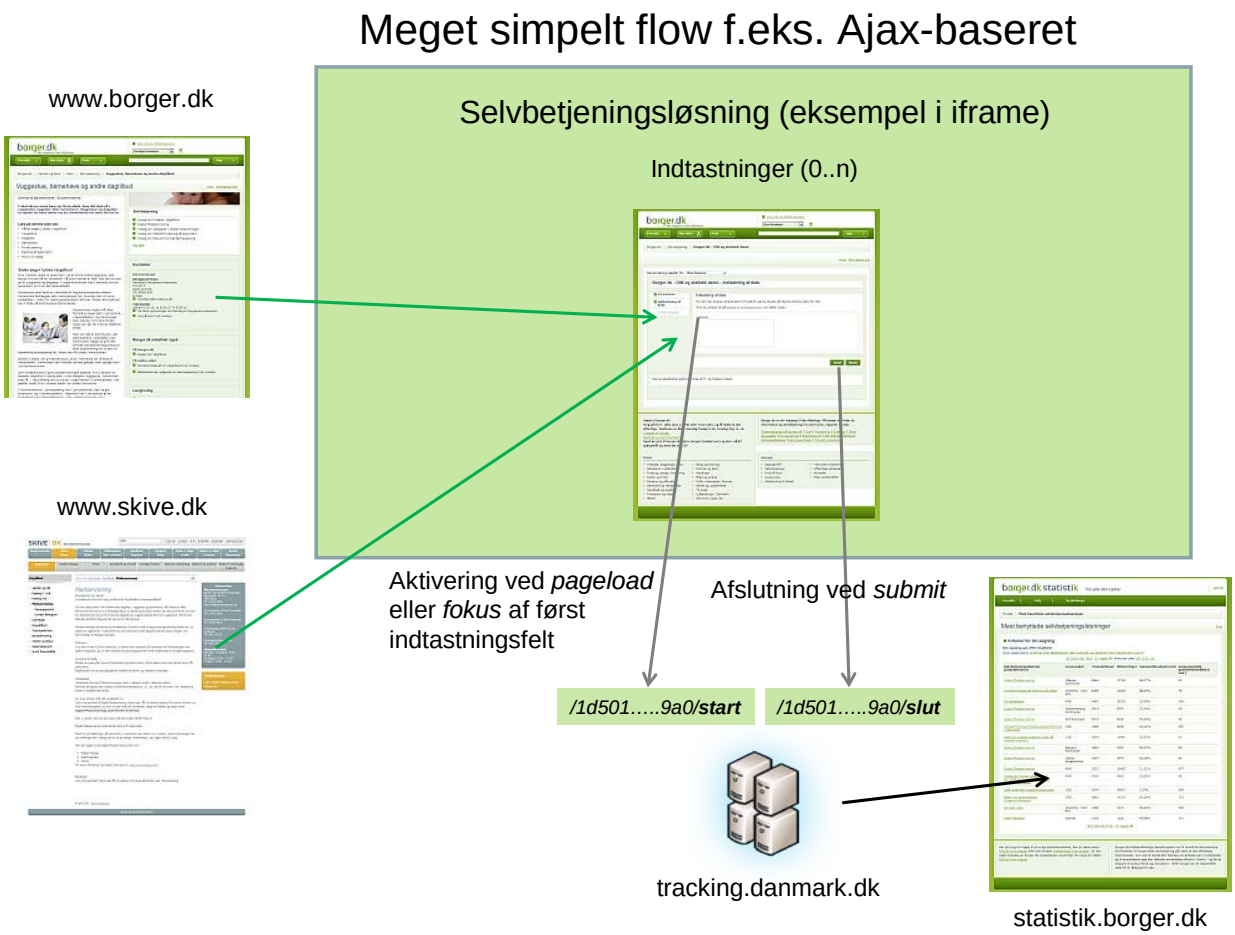
3.4.3 *Komplekst flow – separat start:*

Dette beskriver, hvordan tællerscripts skal implementeres på løsninger, hvori brugeren kan gennemføre flere separate handlinger i én session, fx "Din boligstøtte" hvor brugeren både kan beregne og ansøge. Der placeres startscripts med respektive portalserviceID'er på første trin af hvert af de enkelte selvbetjeningsspor. Der placeres samtidig slutscripts med tilhørende portalserviceID'er på hvert af de mulige afslutnings-spor. Med denne metode sikres den mest korrekte opsamling af data om hvert selvbetjeningsflow fra start til slut. Bemærk at dette forudsætter at de forskellige handlingsspor i løsningen er registreret som selvstændige services hos borger.dk.



3.4.4 *Meget simpelt flow fx. Ajax-baseret:*

Startscript implementeres, så det aktiveres ved indlæsning af siden eller ved konkret indtastning. Slutscript implementeres, så det aktiveres ved "submit", altså når brugeren afslutter selvbetjening med klik på "send" eller lignende. Dette er særligt velegnet til løsninger, som udgøres af simple html-formularer eller lignende.



3.5 Afslutninger, der ikke er en sidevisning

Hvis nærmest tilgængelige afslutning af en brugeraktivitet ikke er en sidevisning, men i stedet er en aktivering af et link til fx en PDF, kan der i stedet laves en tracking af selve klikket.

På en knap eller et link vil man kunne lægge flg. kode:

```
onclick="urchinTracker('/PORTALSERVICEID/slut');"
```

Eller eksempelvis inkl. angivelse af afstigningstrin:

```
onclick="urchinTracker('/PORTALSERVICEID/slut/printPDF');"
```

Dette kræver også, at første del af JavaScript-blokken er inkluderet på siden.

3.6 Anvendelse af leverandørens eget ID

Hvis ikke der benyttes PORTALSERVICEID kan leverandørens eget ID benyttes, hvis det er registeret. Ønskes dette erstatter man PORTALSERVICEID med

```
/SupplierFormat/SupplierID/SuppliersTechnicalSolutionID/MunicipalityCode
```

- SupplierFormat er tekst, der bruges af systemet, for at vide at det er leverandørens eget ID, der skal benyttes.
- SupplierID er leverandørens ID i statistiksystemet, dette kan fås ved at kontakte borger.dk
- SupplierTechnicalSolutionID er leverandørens eget id til servicen, som skal være registreret, før der kan opsamles statistik for løsningen.
- MunicipalityCode er den officielle kommunekode. Sendes ingen kode med opfattes servicen som at gælde for hele landet.

Leverandøren kan vedligeholde eget ID på den enkelte løsning via admin@borger.dk.

4. Tjekliste for installering og validering af udstillet data

Når du installerer tællerscript, skal du kvalitetssikre selve installationen og validere, at der indsamles data korrekt og ikke varierer fra antallet af aktiveringer/starter og gennemførte transaktioner/afslutninger.

4.1 Proces

Test/verifikation af, at tællerscriptet er installeret korrekt, foregår typisk i 3 faser, som beskrevet nedenfor. Er tællerscript allerede implementeret på en eksisterende løsning, bør man starte direkte med fase 3, hvis det er muligt. Kun hvis der er observeret fejl eller hvis det ikke umiddelbart er muligt at lave fase 3, skal de to første faser gennemføres. Generelt er det relativt let teknisk at implementere og teste som en del af udviklingsforløbet.

Fase 1: Under udvikling. Når tællerscriptet indsættes på de relevante sider i løsningen, skal det kontrolleres, at scriptet er indsat og konfigureret på de rigtige sider og afvikles korrekt. Dette gøres ved

- Når en side er renderet, vises det så korrekt i den genererede markup/kildekode
- Laves der et kald med korrekt tæller-ID

Dette er også beskrevet i guiden ”Hvordan kan man se om statistik-scriptet er implementeret korrekt?”

Fase 2: Under test af løsningens samlede funktionalitet. Dette foregår under den integrations- og accepttest, der foregår mellem leverandøren/myndigheden og Digitaliseringsstyrelsen, og består typisk af to forskellige testaktiviteter.

Det angives, at der er tale om testdata ved at tilføje en parameter på tæller-id'et med *test=true*, som fx

*urchinTracker("/aaaaaaaa-bbbb-bbbb-cccc-ddddeeeeffff/slut?**test=true**");*

Oversigt over, hvilke testdata der er modtaget, kan findes via link på forsiden af anvendelse.borger.dk .

Hvis der er behov for hjælp til analyse og fejlsøgning kan man kontakte supporten på admin@digst.dk.

- **Funktion**. Denne test gennemgår alle flows, der er i løsningen, at de fungerer som forventet, herunder at scriptet afvikles. Her testes primært at tællerscriptet findes og afvikles de aftalte steder.
- **Belastning**. Denne test løber et antal af flows igennem med et vist volumen (det anbefales mindst 50) i forhold til samtidighed og vedholdenhed. Her testes, at opgørelsen stemmer overens med de gennemløb, der er lavet.

Fase 3: Verifikation i produktion. En verifikation, hvor opgørelsen sammenlignes med opsamling fra produktionsmiljøet fx logs.

4.2 Checkliste

Her er en kort checkliste som kan bruges

1. Under udvikling

1.1. Se i markup af genereret side:

- at der både er reference til javascript-filen *urchin.js* ?
- at der i separat javascript-blok laves kald til metoden *urchinTracker?*
- at der står det korrekte tæller-ID og hhv */start* eller */slut?*

1.2. Se i netværkstrace (via inspector i browser):

- at der laves kald til *urchin.js?*
- at der laves kald til *tracking.danmark.dk/tracking.gif.aspx* med korrekt værdi af *utmp* parameter?

2. Under test af løsningens samlede funktionalitet

2.1. Funktion:

- Afvikles scriptet korrekt på alle afslutnings-/kvitteringssider?
- Hvis det er muligt at gå tilbage i flowet, er det så sikret, at der kun laves nye afslutninger, hvis der faktisk indsendes data igen?
- Hvis der også registreres aktiveringer, afvikles disse så forud for afslutningerne?

2.2. Belastning:

- Dagen efter, testen er gennemført, kan dataopgørelsen ses under *Testdata*, som ligger adskilt fra produktionsdata.

3. Verifikation efter idriftsættelse. Når løsningen er sat i produktion, skal det sikres:

- At det ikke angives som testdata?
- At der begynder at vises transaktioner i statistikopgørelsen.

Efter at have været i produktion et stykke tid bør der laves en verifikation. Dette gøres for en periode med stabil drift (fx en dag) ved at sammenlige statistikopgørelsen og data fra løsningen (evt. ved gennemgang af logs).

4.3 Forbehold

De data, der vises i statistikopgørelsen, vil være korrekte, men der accepteres en afvigelse op til 5 %. Typisk er det lavere, hvis tællerscriptet er installeret korrekt og verificeret. Baggrunden er, at der er tale om webteknologi, som involverer brugerens browser og adgang via internet/WAN, hvor fejlkilder fx kan være udfald på brugerens netværksadgang, eller at brugeren anvender plugins, som hindrer afviklingen af tællerscriptet.

5. Ændringslog

Ændret den	Version	Ændringsbeskrivelse	Ansvarlig
01-03-2011	2.3		Martin Ege Nielsen, DIGST
01-08-2011	2.3.1		Brian Nielsen, DIGST
01-08-2013	2.4	Justeringer i eksempler. Introduktion af sektion 4. Tjekliste	Brian Nielsen / Morten Meyerhoff Nielsen, DIGST
13-09-2013	2.5	Introduktion af sektion 1.1 (ny sektion) og justeringer sektion 1.2 (gammel sektion 1.1)	Brian Nielsen / Morten Meyerhoff Nielsen, DIGST
16-12-2013	2.5.1	Rettet links i sektion 4. Tjekliste for trin 4 og trin 5.	Morten Meyerhoff Nielsen, DIGST
25-08-2015	2.6	Rettet i afsnit 3.3 Flere mulige afslutninger af en aktivitet. Fjernet afsnit 3.4 Flere separate handlingsspor i en portalservice, som havde en fejl med understøttelse af trin på aktiveringer	Brian Nielsen, DIGST
18-11-2015	2.7	Rettet i flere afsnit. Det er frivilligt, men anbefales at benytte både start-script og slut-script. Slut-script skal dog altid benyttes. Kapitel 4 opdateret med nyeste vejledning til test af implementering	Peter Houmann, DIGST