SimplePay API v2.1 oneclick és tokenes (recurring) fizetés

Fejlesztési dokumentáció

Fizetési folyamat és fejlesztés dokumentáció tárolt kártyás fizetések esetére

2020.08.16.

TARTALOM

Dokumentum történet

Dátum	Verzió	Változás
2019. 02. 22.	190222	Eredeti kiadás
2019. 03. 22.	190322	OneClick fizetések beemelése az általános dokumentációból
2019. 04. 05.	190405	Teszt pontok bővítése, 8. fejezet
2019. 06. 03.	190603	Dokumentáció revízió
2019. 06. 16.	190616	Fordítás előtti revízió
2020. 01 .15.	200115	Erős ügyfélhitelesítés (SCA, 3DS) hozzáadása
2020. 03 .16.	200316	Erős ügyfélhitelesítési folyamat pontosítások - do folyamat - dorecurring folyamat
2020. 04 .07.	200407	Új folyamatok - Egyszerűsített kétlépcsős kártyaregisztrációs folyamat
2020. 05. 29.	200529	Új folyamatok - do – API v2 fizetés API v1 használatával tárolt kártyával - do – API v2 fizetés OTP Webshop (Middleware, MW) szolgáltatásban tárolt kártyával - tokenquery - token állapotának lekérdezése - tokencancel - token inaktiválása
2020. 08. 16.	200816	Pontosítások - Erős ügyfélhitelesítés - threeDSReqAuthMethod bevezetése Új folyamatok - Legacy oneclick és recurring kártyatárolás

1 Rövid összefoglalás

1.1 A SimplePay online fizetés v2.0 és v2.1

A SimplePay az OTP Mobil Kft. online fizetési megoldása. Jelen dokumentáció a SimplePay fizetéshez használható kereskedői API és tranzakció kezelés v2 verzióján alapuló v2.1 kártyatároláshoz és fizetéshez készült.

A leírás háromféle kártyatároláson alapuló üzleti logikát és technikai megvalósítást mutat be:

- **oneclick**: a vásárló jelenlétével történő ismétlődő fizetések
- **recurring**: tokenes, automatikusan ismétlődő fizetések
- **legacy**: az OTP Bank rendszerben történő kártyatárolás oneclick és recurring esetére is

A fenti témakörök korábban külön dokumentációkban kaptak helyet, azonban a továbbiakban ez a közös dokumentáció tárgyalja a SimplePay rendszer (v2.1 API) összes kártyatárolási megoldását.

Jelen dokumentáció nem tárgyalja a vásárló által a SimplePay fizetőoldalon kiválasztható „Simple" kártyatárolást, mivel az nem igényel kereskedő oldali fejlesztést.

Jelen dokumentáció csak a fent megnevezett ismétlődő fizetések esetre nyújt tájékoztatást, de az alapokat és a kártyatárolás nélküli fizetés folyamatát nem tárgyalja.

Jelen dokumentáció az erős ügyfélhitelesítést **(PSD2, SCA, 3DS)** csak a tárolt kártyás fizetésekkel kapcsolatban tárgyalja.

Ezekből adódóan a továbbiak az általános SimplePay fizetési dokumentáció ismerete nélkül nem értelmezhetők!

Az erős ügyfélhitelesítés technikai megoldása a tárolt kártyás fizetések esetére aktív fejlesztés alatt áll. Emiatt a dokumentáció is gyakrabban bővülhet a meglevő folyamatok további leírásával, vagy új funkciók bevezetése esetén.

Az általános SimplePay technikai dokumentáció aktuális változata a kereskedői admin felület (sandbox és éles rendszeren) letöltési szekciójában, vagy az alábbi URL-en érhető el:

Magyarul: http://simplepartner.hu/download.php?target=v21dochu
Angolul: http://simplepartner.hu/download.php?target=v21docen

Jelen dokumentáció minden ponton a fenti általános dokumentációra hivatkozik, vagy annak ismeretét feltételezi, amikor az adott témakört tárgyalja.

1.2 Élesítési feltételek

Minden élesítés előtt álló kereskedői rendszer (webáruház, mobil applikáció) tesztelésen esik át. A kártyatárolás alkalmazása esetén ez abban az esetben is megtörténik, ha egy már korábban tesztelt rendszer lett kiegészítve ezzel a funkcióval.

Kártyatárolás esetén - az alapvető fizetési teszteken túl – jelen dokumentáció „**Tesztelés**" fejezetében leírtak ellenőrzése is szükséges.

1.3 Fogalmak

Kártya regisztráció: a bankkártyás fizetés opciója, ami során a fizetéshez használt kártya adatai el lesznek mentve. a SimplePay rendszerében Csak sikeres authorizációhoz tartozó kártya adatai menthetők.
PCI-DSS (Payment Card Industry Data Security Standard): bankkártya tárolásra vonatkozó adatbiztonsági szabvány. A SimplePay rendszerében a kártyák tárolása a PCI-DSS szabályainak megfelelően, folyamatosan auditált környezetben történnek.
cardSecret: a vásárló által a regisztrációs tranzakciónál megadott egyedi, csak általa ismert adat. A továbbiakban minden olyan tranzakciónál meg kell adja, amit az elmentett kártyájával indít, ezzel biztosítva a jelenlétét.
OneClick fizetés: a **cardSecret** eseti felhasználásával a vásárló kezdeményezi a fizetést. Alapvetően kényelmi szolgáltatás, mert a továbbiakban nem szükséges a fizetőoldalon történő kártyaszám megadás.
Token: a regisztrációs fizetés esetén generált azonosító, ami használatával későbbi fizetések kezdeményezhetők az elmentett kártyával.
Recurring fizetés: a kereskedő által a kártya tulajdonosának jelenléte nélkül, automatikusan kezdeményezett ismétlődő fizetések a **token** használatával, pl. előfizetéses szolgáltatásoknál
Legacy kártyatárolás: Az OTP Bank rendszerében történő kártyatárolás oneclick és recurring esetére.

1.4 Oneclick fizetés

A oneclick fizetés lehetőséget ad arra, hogy a vásárló a tranzakció során elmenthesse a bankkártyáját. A következő vásárlásnál már nem kell átirányítani a SimplePay fizető oldalára, hanem elegendő a háttérben elindítani a tranzakciót, így a vásárló végig a kereskedői weboldalon marad. Ez a kártyatárolásos fizetés csak abban az esetben alkalmazható, ha a vásárló jelen van (user present) a vásárlásnál és a fizetési tranzakciónál.

1.5 Oneclick fizetési tranzakciók folyamata

1.5.1 Kártyaregisztrációs fizetés oneClick esetén

Az első tranzakció során történik meg a vásárló kártyájának regisztrációja. Az első, kártyaregisztrációs tranzakció során a fizetőoldalon történik meg a **3DS** ellenőrzés, így ezzel a kereskedői rendszernek csak annyi feladata van, hogy a szükséges vásárlói adatokat elküldje a „**start**" hívásban.

a. A vásárló a kereskedő weboldalán összeválogatja a termékeket, ami eredményeképpen kialakul a fizetendő végösszeg.
b. A kereskedő lehetőséget biztosít arra, hogy a weboldalon a vásárló egy egyedi azonosítót (**cardSecret**) regisztrálhasson be a SimplePay rendszerében. A tranzakció indításakor ezzel az adattal kell kiegészíteni a start funkciót. A cardSecret értékét a kereskedői rendszer nem mentheti, csak továbbíthatja a SimplePay felé. A későbbiekben indítandó oneClick fizetéseknél csak a cardSecret vásárló általi megadásával lehet a tranzakciót elindítani.
c. A tranzakciós adatokat a kereskedő átadja a SimplePay felé az API-n keresztül (**start**). Ezen a ponton létrejön a fizetési tranzakció a SimplePay rendszerében. A kapott

adatokra válaszként a rendszer egy URL-t ad vissza. A kereskedő erre az URL-re kell irányítsa a vásárlót a böngészőben.

d. A megadott URL-en a SimplePay fizetőoldalra érkezik a vásárló, ahol a tranzakció korábban megadott adatai jelennek meg. A fizetőoldalon tudja megadni a vásárló a kártyájának adatait. Ha van a Simple applikációban regisztrált kártája, akkor azt választva is elvégezheti a fizetést.
e. A kártyaadatok megadása után megtörténik a fizetés banki hitelesítése (authorizáció).
f. Az authorizáció után a vásárló vissza van irányítva a kereskedő weboldalára (**back**). Itt a visszaadott adatok alapján szükséges tájékoztatni a vásárlót a tranzakció eredményéről.
g. Ezután a háttérben lefut a csalás megfigyelő és megelőző folyamat. Ha ennek során nem észlel a rendszer semmilyen problémát, akkor a háttérben visszajelzést küld a weboldal felé (**ipn**). Ez a fizetési tranzakció vége. Miután az IPN üzenet megérkezik, teljesítheti a megrendelést a kereskedő.
h. Az IPN üzenet tartalmazza:

- a regisztrált kártya lejárati dátumát
- a kimaszkolt kártyaszámot, ahol az utolsó 4 számjegy olvasható

1.5.2 Regisztrált kártyás fizetés oneClick esetén (a kártyatulajdonos jelenlétével)

A már regisztrált kártyával indíthatók olyan tranzakciók, ahol a kártyaadatok megadása nélkül is lehet fizetni

a. A vásárló a kereskedő weboldalán összeválogatja a termékeket, ami eredményeképpen kialakul a fizetendő végösszeg.
b. A kereskedő lehetőséget biztosít arra, hogy a weboldalon a vásárló korábban a kártyaregisztrációhoz használt egyedi azonosítót (**cardSecret**) megadhassa. A cardSecret értékét a kereskedői rendszer nem mentheti, csak továbbíthatja a SimplePay felé.
c. A vásárlási adatokat a cardSecret értékével kiegészítve a háttérben átadja a SimplePay felé az API-n keresztül (**do**). A vásárló eközben a webáruház oldalán marad, nincs szükség átirányításra.
d. A SimplePay rendszere **elküldi a bank felé** a vásárlási adatokat **3DS ellenőrzésre**.
e. Ha a **banki 3DS** ellenőrzés **sikertelen**, akkor
   - Az authorizáció ezen a ponton **nem** történik meg.
   - Ebben az esetben a **do** hívás válaszában a SimplePay rendszere visszaad egy URL-t a kereskedőnek, amire átirányíthatja a vásárlót (**challenge**).
   - Ha ezután a kereskedő átirányítja a kapott URL-re a vásárlót, akkor egy olyan helyre jut, ahol a kártyakibocsátó bank által megkívánt további ügyfélellenőrzési folyamat történik meg.
   - Ha a fentiek után az ellenőrzés, majd a fizetés sikeres eredménnyel zárul, akkor a háttérben lefut a csalás megfigyelő és megelőző folyamat. Ha ennek során nem észlel a rendszer semmilyen problémát, akkor a háttérben visszajelzést küld a weboldal felé (**ipn**). Ez a fizetési tranzakció vége. Miután az IPN üzenet megérkezik, teljesítheti a megrendelést a kereskedő.
f. Ha a **banki 3DS** ellenőrzés **sikeres**, akkor a SimplePay rendszere beazonosítja a mentett kártyát, ami után megtörténik a fizetés banki hitelesítése (authorizáció). Sikeresség esetén tehát **nincs challenge**, így a fizetési folyamat a háttérben tud futni.

g. A SimplePay rendszere szinkron választ ad a **do** hívásra az authorizáció eredményéről. Logikailag ez a pont megfelel a normál fizetés esetén a kereskedői weboldalra való visszairányítással (**back**).
h. Ezután a háttérben lefut a csalás megfigyelő és megelőző folyamat. Ha ennek során nem észlel a rendszer semmilyen problémát, akkor a háttérben visszajelzést küld a weboldal felé (**ipn**). Ez a fizetési tranzakció vége. Miután az IPN üzenet megérkezik, teljesítheti a megrendelést a kereskedő.

1.6 Recurring fizetés

A recurring fizetések (ismétlődő fizetések) alatt azt értjük, amikor egy regisztrált kártyával a kártya tulajdonos jelenléte nélkül lehet több alkalommal fizetéseket indítani. Ez a fizetési folyamat abban az esetben hasznos, ha automatikusan szeretne a kereskedő fizetéseket indítani, például havi díjak beszedésére.

A recurring és a korábban tárgyalt OneClick folyamat egymástól technikailag is különbözik. Emiatt meg kell különböztetni azokat a regisztrált kártyás fizetéseket, amikor a kártyatulajdonos jelenlétével (oneclick) történik meg a fizetés elindítása és amikor a jelenléte nélkül (recurring).

A recurring kártyaregisztrációt kétféle folyamatban lehet elvégezni a kereskedő üzleti logikájának megfelelően.
- egylépcsős, azonnali terheléssel (pl. egy vásárlás részeként)
- kétlépcsős, terhelés nélkül (csak a regisztráció a cél)

1.7 Recurring fizetési tranzakciók folyamata

1.7.1 Egylépcsős kártyaregisztrációs folyamat

Ebben az esetben egy valós vásárlás és kártyaterhelés is történik, aminek paramétereként egyben tokeneket is kér a kereskedő a SimplePay rendszerétől. A tokenekkel lehet majd a későbbiekben a vásárló jelenléte nélkül tranzakciókat indítani.

Az első, kártyaregisztrációs tranzakció során a fizetőoldalon történik meg a **3DS** ellenőrzés, így ezzel a kereskedői rendszernek csak annyi feladata van, hogy a szükséges vásárlói adatokat elküldje a „**start**" hívásban.

i. A vásárló a kereskedő weboldalán összeválogatja a termékeket, ami eredményeképpen kialakul a fizetendő végösszeg.
i. A kereskedő (a vásárló egyetértésével) a tranzakció elindítása előtt meg kell határozza, az igényelt tokenek paramétereit az alábbiak szerint:
    a. darabszám (1-24), ennyi különálló további fizetést lehet majd indítani a regisztrált kártyán
    b. maximális értékhatár, maximum ekkora összegű tranzakciókat lehet indítani a későbbiekben a tokenekkel
    c. lejárat ideje, maximum eddig a határidőig lehet fizetést kezdeményezni a generált tokenekkel.
j. A tranzakciós adatokat a token paraméterekkel együtt a kereskedő átadja a SimplePay felé az API-n keresztül (**start**). Ezen a ponton létrejön a fizetési tranzakció a

SimplePay rendszerében. A kapott adatokra válaszként a rendszer egy URL-t ad vissza. A kereskedő erre az URL-re kell irányítsa a vásárlót a böngészőben.

k. A start hívás válaszában találhatók meg a generált tokenek is. A tokenek csak az authorizáció sikeressége után lesznek aktívak. A tokeneket le kell tárolja a kereskedő, hogy a későbbi fizetésekhez fel lehessen majd használni.
l. A **c.** pontban megadott URL-en a SimplePay fizetőoldalra érkezik a vásárló, ahol a tranzakció korábban megadott adatai jelennek meg. A vásárlás részletei mellett fel vannak tüntetve a **b**. pontban megadott token paraméterek is, így a vásárló azt ellenőrizheti. A vásárló a fizetőoldalon tudja megadni a kártyájának adatait.
m. A kártyaadatok megadása után megtörténik a fizetés banki hitelesítése (authorizáció).
n. Az authorizáció után a vásárló vissza van irányítva a kereskedő weboldalára (**back**). Itt a visszaadott adatok alapján szükséges tájékoztatást adni az authorizáció eredményéről.
o. Ezután a háttérben lefut a csalás megfigyelő és megelőző folyamat. Ha ennek során nem észlel a rendszer semmilyen problémát, akkor a háttérben visszajelzést küld a weboldal felé (**ipn**). Ez a fizetési tranzakció vége. Miután az IPN üzenet megérkezik, teljesítheti a megrendelést a kereskedő.
p. Az IPN üzenet tartalmazza:
    a. a regisztrált kártya lejárati dátumát
    b. a kimaszkolt kártyaszámot, ahol az utolsó 4 számjegy olvasható
q. A kért tokenek az IPN üzenettel egyidőben aktiválódnak. A tokenek a tranzakcióhoz regisztrált kártyával történő fizetésekhez használhatók fel a későbbiekben.

1.7.2 Kétlépcsős kártyaregisztrációs folyamat

Ebben az esetben a regisztráció során csak egy jelképes, pl. 100 forintos authorizáció történik, ami csak befoglalásra kerül a vásárló számláján, majd ennek sikeressége esetén ez azonnal fel van szabadítva. A hívás célja csak a regisztráció és a tokenek generálása. A valós vásárlások a kapott tokenekkel lesznek elindítva a kereskedő által.

Az első, kártyaregisztrációs tranzakció során a fizetőoldalon történik meg a **3DS** ellenőrzés, így ezzel a kereskedői rendszernek csak annyi feladata van, hogy a szükséges vásárlói adatokat elküldje a „**start**" hívásban.

A kereskedői fiókon a kétlépcsős fizetésnek aktívnak kell legyen a funkció használata előtt. A funkció fejlesztéséhez elengedhetetlen a hivatkozott alap dokumentációban leírt kétlépcsős fizetések logikájának ismerete.

a. A vásárló a kereskedő weboldaláról elindít agy jelképes, a kereskedő által megadott összegű, pl. 100 forintos tranzakciót.
b. A kereskedő a tranzakció elindítása előtt meg kell határozza, a tokenek paramétereit az alábbiak szerint:
    a. darabszám (1-24), ennyi különálló további fizetést lehet majd indítani a regisztrált kártyán
    b. maximális értékhatár, maximum ekkora összegű tranzakciókat lehet indítani a későbbiekben a tokenekkel
    c. lejárat ideje, maximum eddig a határidőig lehet fizetést kezdeményezni a generált tokenekkel.
c. A tranzakciós adatokat a token paraméterekkel együtt a kereskedő átadja a SimplePay felé az API-n keresztül (**start**). Ezen a ponton létrejön a fizetési tranzakció a

SimplePay rendszerében. A kapott adatokra válaszként a rendszer egy URL-t ad vissza. A kereskedő erre az URL-re kell irányítsa a vásárlót a böngészőben.

d. A start hívás válaszban találhatók meg a generált tokenek is. A tokenek csak az authorizáció sikeressége után lesznek aktívak. A tokeneket le kell tárolja a kereskedő, hogy a későbbi fizetésekhez fel lehessen majd használni.
e. A **c.** pontban megadott URL-en a SimplePay fizetőoldalra érkezik a vásárló, ahol a tranzakció korábban megadott adatai jelennek meg. A vásárlás részletei mellett fel vannak tüntetve a **b**. pontban megadott token paraméterek is, így a vásárló azt ellenőrizheti. A vásárló a fizetőoldalon tudja megadni a kártyájának adatait.
f. A kártyaadatok megadása után megtörténik a fizetés banki hitelesítése (authorizáció). **Kétlépcsős fizetés esetén az authorizáció során csak a megadott összeg befoglalása történik meg, de terhelés nem. A tranzakció lezárásához (terhelés, vagy feloldás) további kereskedői interakcióra van szükség.**
g. Az authorizáció után a vásárló vissza van irányítva a kereskedő weboldalára (**back**). Itt a visszaadott adatok alapján szükséges tájékoztatást adni az authorizáció eredményéről.
h. Ezután a háttérben lefut a csalás megfigyelő és megelőző folyamat. Ha ennek során nem észlel a rendszer semmilyen problémát, akkor a háttérben visszajelzést küld a weboldal felé (**ipn**). Az IPN üzenet itt a sikeres authorizációról értesít. Ez az IPN üzenet opcionális a kétlépcsős fizetések esetén.
i. Az IPN üzenet tartalmazza:
    a. a regisztrált kártya lejárati dátumát
    b. a kimaszkolt kártyaszámot, ahol az utolsó 4 számjegy olvasható
j. A kért tokenek az IPN üzenettel egyidőben aktiválódnak. A tokenek a tranzakcióhoz regisztrált kártyával történő fizetésekhez használhatók fel a későbbiekben.
k. Az IPN üzenet után elindítható a befoglalt összeg felszabadítása (**finish**). A finish hívás a létlépcsős tranzakció lezárására szolgál. Ha 0 összeggel van indítva, akkor ennek hatására a teljes befoglalt összeg ismét elérhető lesz a kártyatulajdonos számára. Ezzel a szükséges mennyiségű aktív token létrejött anélkül, hogy a vásárló kártyáján valós terhelés történt volna.
l. A finish hívás után is IPN üzenetet küld a SimplePay rendszere a kereskedőnek, ami a tranzakció végső összegét tartalmazza, ami jelen esetben 0 érték. Ez az IPN üzenet nem opcionális.

1.7.3 Egyszerűsített kétlépcsős kártyaregisztrációs folyamat

Ebben az esetben a kereskedői rendszer már a „**start**" hívásban előre meghatározhatja (**onlyCardReg**), hogy a tranzakció célja kizárólag a kártyaregisztráció, azaz a tranzakció összegét nem akarja majd terhelni.

Kétlépcsős fizetés esetén ez annyit jelent, hogy a szükséges összeg sikeres befoglalása után a rendszer automatikusan elvégzi a teljes összeg feloldását is, így a kereskedői rendszernek nem szükséges egy újabb API hívásban elindítani ezt a folyamatot.

A fentebb leírt kétlépcsős kártyaregisztrációs folyamatnál annyival egyszerűbb így, hogy ebben az esetben a teljes „**finish**" hívás elhagyható.

1.7.4 Regisztrált kártyás fizetés recurring esetén

A korábban generált tokennel indított fizetésnél nincs jelentősége annak, hogy a regisztráció egy-, vagy kétlépcsős módon történt. Ezen a ponton csak az fontos, hogy a regisztrációs tranzakció során sikeres authorizáció történjen, ami hatására aktiválódtak a generált tokenek. Ugyanakkor az szabályozható, hogy a tokennel indított fizetés egy-, vagy kétlépcsős módon induljon el.

a. A kereskedő rendszerében elérkezik az ideje egy fizetésnek, például egy rendszeres szolgáltatás havidíjának kiegyenlítésére. Ennek során kialakul a fizetendő végösszeg.

j. A kereskedő a vásárlóhoz letárolt tokenek közül kiválaszt egyet, ami felhasználásával terhelhető a mentett kártya.

k. A kiválasztott **tokennel** korábban nem történhetett sikeres vásárlás.

l. A vásárlásnak **összegének** a token kérésnél megadott értékhatárnak meg kell feleljen.

m. A vásárlás **időpontjának** a token kérésnél megadott határidőn belül kell legyen.

n. A vásárlási adatokat a tokennel kiegészítve a kereskedő a háttérben átadja a SimplePay felé az API-n keresztül (**dorecurring**).

o. Kétlépcsős fizetésre beállított fiók esetén a dorecurring hívás **opcionálisan** kiegészíthető a **twoStep** változóval.
   - ha a **twoStep** változó **nincs átadva**, akkor a fiók beállításának megfelelően kétlépcsős fizetésként fog futni a tokenes fizetés is, azaz egy „**finish**" hívással kell lezárni.
   - ha a **twoStep** változó **át van adva** és az értéke **true**, akkor is a fiók beállításainak megfelelően kétlépcsős fizetésként fog futni a tokenes fizetés, azaz egy „**finish**" hívással kell lezárni.
   - ha a **twoStep** változó **át van adva** és az értéke **false**, akkor a **„finish" hívás nélkül** is el fog indulni **a teljes összegű terhelés** a sikeres authorizáció után.

- A SimplePay rendszere **elküldi a bank felé** a vásárlási adatokat **3DS ellenőrzésre**.
- Ha a **banki 3DS** ellenőrzés **sikertelen**, akkor
  - Az authorizáció ezen a ponton **nem** történik meg.
  - Ebben az esetben a **dorecurring** hívás válaszában a SimplePay rendszere visszaad egy olyan URL-t a kereskedőnek, amire átirányíthatja a vásárlót (**challenge**).
  - Ha ez egy teljesen automatikus fizetési kezdeményezés volt, ahol a vásárló nincs jelen, emiatt nem is lehet átirányítani, akkor ez a fizetési folyamat itt lezárul a **sikertelen 3DS** ellenőrzés **miatt**.
  - Ha a vásárló jelen van a fizetésnél, és a kereskedő át tudja irányítani a kapott URL-re, akkor egy olyan helyre jut, ahol a kártyakibocsátó bank által megkívánt további ügyfélellenőrzési folyamat történik meg.
  - Ha a fentiek után az ellenőrzés, majd a fizetés sikeres eredménnyel zárul, akkor a háttérben lefut a csalás megfigyelő és megelőző folyamat. Ha ennek során nem észlel a rendszer semmilyen problémát, akkor a háttérben visszajelzést küld a weboldal felé (**ipn**). Ez a fizetési tranzakció vége. Miután az IPN üzenet megérkezik, teljesítheti a megrendelést a kereskedő.

r. Ha a **banki 3DS** ellenőrzés **sikeres,** akkor a SimplePay rendszere beazonosítja a mentett kártyát, ami után megtörténik a fizetés banki hitelesítése (authorizáció).

p. A SimplePay rendszere szinkron választ ad a **dorecurring** hívásra az authorizáció eredményéről. Logikailag ez a pont megfelel a normál fizetés esetén a kereskedői weboldalra való visszairányítással (**back**).

q. Ezután a háttérben lefut a csalás megfigyelő és megelőző folyamat. Ha ennek során nem észlel a rendszer semmilyen problémát, akkor a háttérben visszajelzést küld a weboldal felé (**ipn**).
r. Egylépcsős fizetés esetén ez a fizetési tranzakció vége. Miután az IPN üzenet megérkezik, teljesítheti a megrendelést a kereskedő.
s. Kétlépcsős fizetés esetén a „**finish**" hívással kell lezárni a tranzakciót. Ez alól kivétel a fentebb említett **twoStep** változó **false** értékkel való használata.
t. A token egy sikeres fizetéshez használható fel. Sikeresség esetén felhasználttá válisk és több alkalommal már nem indítható vele fizetés.

1.7.5 Tokenek

A start hívásban generált tokeneket a kereskedői rendszerben le kell tárolni. A későbbiekben ezekkel lehet tranzakciót indítani. A tokenek az alábbi jellemzőkkel bírnak:

- A tokenek egy konkrét sikeres, kártyaregisztrációs SimplePay bankkártyás fizetéshez tartoznak.
- A tokenekkel indítható tranzakciók a sikeres fizetésnél felhasznált bankkártyát fogják terhelni.
- Adott regisztrációhoz tartozó tokenek száma nem bővíthető. Ha további tokenekre van szükség, akkor ahhoz egy újabb kártyaregisztrációs tranzakciót szükséges indítani.
- A kártya regisztrációja addig érvényes, ameddig
    - az összes hozzá generált token nincs elhasználva
    - **és** a generálásnál megadott lejárati idő még nem múlott el
    - **és** a regisztrált bankkártya érvényességi ideje nem múlott el
- A fenti két időpont közül minden esetben a közelebbi az érvényesség határideje.
- A SimplePay rendszere abban az esetben tudja az autorizációt elindítani a tokenes kérésre, ha
    - a fenti, a regisztráció érvényességére vonatkozó kritériumok teljesülnek
    - a tranzakció összege a token maximális értékhatárát nem lépi túl
    - a tokennel még nem történt sikeres tranzakció
    - a tranzakció indítása technikailag megfelelő módon történik
- A tokennel indított tranzakcióra a terhelés a banki oldalon akkor lesz végrehajtva, ha az adott pillanatban a korábban regisztrált bankkártya
    - érvényes (nem járt le, nincs letiltva, nincs elvesztve, stb)
    - a tranzakcióban megadott összeg rendelkezésre áll a bankkártyán
    - nincs túllépve valamilyen a bankkártya használatára vonatkozó banki beállítás (tranzakciós limitösszeg; napi, havi összeg, vagy darabszám limit, stb.)
- A generált tokenek egyenértékűek
- A tokenek tetszőleges sorrendben használhatók fel
- Mindegyik token csak különálló fizetéshez használható fel.
- Egymással nem vonhatók össze a tokenek
- Sikeres fizetés után a terhelt összegtől függetlenül a token inaktív lesz.
- Csak olyan tokennel lehet fizetést indítani, amivel korábban még nem volt sikeres tranzakció végrehajtva.
- Sikertelen fizetés után (tehát ha nem történt terhelés) a token még felhasználható újabb fizetés kezdeményezéséhez
- A tokenek nem módosíthatók
- A kártyaregisztráció inaktiválásával (**cardcancel**) az összes hozzá tartozó token is inaktív lesz.
- Az inaktiválás nem vonható vissza. Az egyszer inaktívált regisztráció addig felhasználatlan tokenjei a továbbiakban már nem alkalmasak fizetés indításához.
- A kártyával indított fizetések lekérdezhetők (**cardquery**) az inaktiválás, vagy lejárat után is.

1.8 Legacy kártyaregisztráción alapuló fizetések

Ezen a módon technikai szempontból egyszerűbbek lehetnek a regisztrációs és fizetési folyamatok, azonban nem a SimplePay saját rendszerében történik meg a kártya tárolása, hanem az OTP Bank rendszerében.

A tárolás helyének annyiban van jelentősége, hogy a SimplePay saját rendszerében tárolt kártyák esetén több elfogadó felé tudja indítani authorizációra a tranzakciókat attól függően, hogy adott pillanatban melyik érhető el. A banki rendszerben mentett kártyák esetén csak a Bank rendszerében történhet meg fizetéskor az authorizáció.

1.8.1 Legacy oneclick regisztráció

Az első tranzakció során történik meg a vásárló kártyájának regisztrációja. Az első, kártyaregisztrációs tranzakció során a fizetőoldalon történik meg a **3DS** ellenőrzés, így ezzel a kereskedői rendszernek csak annyi feladata van, hogy a szükséges vásárlói adatokat elküldje a „**start**" hívásban.

s. A vásárló a kereskedő weboldalán összeválogatja a termékeket, ami eredményeképpen kialakul a fizetendő végösszeg.
t. A kereskedő **legacy** kártyatárolást oneclick fizetés esetén a **legacyCardRegister** változó használatával jelzi.
u. A tranzakciós adatokat a kereskedő átadja a SimplePay felé az API-n keresztül (**start**). Ezen a ponton létrejön a fizetési tranzakció a SimplePay rendszerében. A kapott adatokra válaszként a rendszer egy URL-t ad vissza. A kereskedő erre az URL-re kell irányítsa a vásárlót a böngészőben.
v. A megadott URL-en a SimplePay fizetőoldalra érkezik a vásárló, ahol a tranzakció korábban megadott adatai jelennek meg. A fizetőoldalon tudja megadni a vásárló a kártyájának adatait.
w. A kártyaadatok megadása után megtörténik a fizetés banki hitelesítése (authorizáció).
x. Az authorizáció után a vásárló vissza van irányítva a kereskedő weboldalára (**back**). Itt a visszaadott adatok alapján szükséges tájékoztatni a vásárlót a tranzakció eredményéről.
y. Ezután a háttérben lefut a csalás megfigyelő és megelőző folyamat. Ha ennek során nem észlel a rendszer semmilyen problémát, akkor a háttérben visszajelzést küld a weboldal felé (**ipn**). Ez a fizetési tranzakció vége. Miután az IPN üzenet megérkezik, teljesítheti a megrendelést a kereskedő.
z. Az IPN üzenet tartalmazza:
- a regisztrált kártya lejárati dátumát
- a kimaszkolt kártyaszámot, ahol az utolsó 4 számjegy olvasható

1.8.2 Legacy oneClick fizetés

A már regisztrált kártyával indíthatók olyan tranzakciók, ahol a kártyaadatok megadása nélkül is lehet fizetni

u. A vásárló a kereskedő weboldalán összeválogatja a termékeket, ami eredményeképpen kialakul a fizetendő végösszeg.
v. A vásárlási adatokat a háttérben átadja a SimplePay felé az API-n keresztül (**do**). A vásárló eközben a webáruház oldalán marad, nincs szükség átirányításra.

w. A vásárlási adatokkal együtt a kereskedő átadja a **type** mezőt is, amiben jelzi, hogy a vásárló jelen van-e a tranzakciónál.
x. A SimplePay rendszere **elküldi a bank felé** a vásárlási adatokat **3DS ellenőrzésre**.
y. Ha a **banki 3DS** ellenőrzés **sikertelen**, **de jelen van a vásárló** akkor
   - Az authorizáció ezen a ponton **nem** történik meg.
   - Ebben az esetben a **do** hívás válaszában a SimplePay rendszere visszaad egy URL-t a kereskedőnek, amire átirányíthatja a vásárlót (**challenge**).
   - Ha ezután a kereskedő átirányítja a kapott URL-re a vásárlót, akkor egy olyan helyre jut, ahol a kártyakibocsátó bank által megkívánt további ügyfélellenőrzési folyamat történik meg.
   - Ha a fentiek után az ellenőrzés, majd a fizetés sikeres eredménnyel zárul, akkor a háttérben lefut a csalás megfigyelő és megelőző folyamat. Ha ennek során nem észlel a rendszer semmilyen problémát, akkor a háttérben visszajelzést küld a weboldal felé (**ipn**). Ez a fizetési tranzakció vége. Miután az IPN üzenet megérkezik, teljesítheti a megrendelést a kereskedő.
z. Ha a **banki 3DS** ellenőrzés **sikertelen**, **és nincs jelen a vásárló** akkor a tranzakció itt sikertelenül ér véget, ugyanis ebben az esetben nem lehet challenge folyamatot indítani.
aa. Ha a **banki 3DS** ellenőrzés **sikeres**, akkor a SimplePay rendszere beazonosítja a mentett kártyát, ami után megtörténik a fizetés banki hitelesítése (authorizáció). Sikeresség esetén tehát **nincs challenge**, így a fizetési folyamat a háttérben tud futni.
bb. A SimplePay rendszere szinkron választ ad a **do** hívásra az authorizáció eredményéről. Logikailag ez a pont megfelel a normál fizetés esetén a kereskedői weboldalra való visszairányítással (**back**).
cc. Ezután a háttérben lefut a csalás megfigyelő és megelőző folyamat. Ha ennek során nem észlel a rendszer semmilyen problémát, akkor a háttérben visszajelzést küld a weboldal felé (**ipn**). Ez a fizetési tranzakció vége. Miután az IPN üzenet megérkezik, teljesítheti a megrendelést a kereskedő.

1.8.3 Legacy recurring regisztráció

Az első tranzakció során történik meg a vásárló kártyájának regisztrációja. Az első, kártyaregisztrációs tranzakció során a fizetőoldalon történik meg a **3DS** ellenőrzés, így ezzel a kereskedői rendszernek csak annyi feladata van, hogy a szükséges vásárlói adatokat elküldje a „**start**" hívásban.

Ebben az esetben is lehetséges a kétlépcsős fizetés a normál **recurring** folyamatnál részletezett módon, illetve az **onlyCadrReg** használata is.

aa. A vásárló a kereskedő weboldalán összeválogatja a termékeket, ami eredményeképpen kialakul a fizetendő végösszeg.
bb. A kereskedő **legacy** kártyatárolást recurring fizetés esetén a **legacyRecurring** változó használatával jelzi.
cc. A tranzakciós adatokat a kereskedő átadja a SimplePay felé az API-n keresztül (**start**). Ezen a ponton létrejön a fizetési tranzakció a SimplePay rendszerében. A kapott adatokra válaszként a rendszer egy URL-t ad vissza. A kereskedő erre az URL-re kell irányítsa a vásárlót a böngészőben.
dd. A megadott URL-en a SimplePay fizetőoldalra érkezik a vásárló, ahol a tranzakció korábban megadott adatai jelennek meg. A fizetőoldalon tudja megadni a vásárló a

kártyájának adatait. Ha van a Simple applikációban regisztrált kártája, akkor azt választva is elvégezheti a fizetést.

ee. A kártyaadatok megadása után megtörténik a fizetés banki hitelesítése (authorizáció).

ff. Az authorizáció után a vásárló vissza van irányítva a kereskedő weboldalára (**back**). Itt a visszaadott adatok alapján szükséges tájékoztatni a vásárlót a tranzakció eredményéről.

gg. Ezután a háttérben lefut a csalás megfigyelő és megelőző folyamat. Ha ennek során nem észlel a rendszer semmilyen problémát, akkor a háttérben visszajelzést küld a weboldal felé (**ipn**). Ez a fizetési tranzakció vége. Miután az IPN üzenet megérkezik, teljesítheti a megrendelést a kereskedő.

hh. Az IPN üzenet tartalmazza:

- a regisztrált kártya lejárati dátumát
- a kimaszkolt kártyaszámot, ahol az utolsó 4 számjegy olvasható
-

1.8.4 Legacy recurring fizetés

a. A kereskedő rendszerében elérkezik az ideje egy fizetésnek, például egy rendszeres szolgáltatás havidíjának kiegyenlítésére. Ennek során kialakul a fizetendő végösszeg.
b. A vásárlási adatokat a háttérben átadja a SimplePay felé az API-n keresztül (**do**).
c. A vásárlási adatokkal együtt a kereskedő átadja a **legacyRecurring** változót, amivel jelzi, hogy ez egy recurring tranzakció, illetve a **type** mezőt is, amiben jelzi, hogy a felhasználó nincs jelen a tranzakciónál.
d. A SimplePay rendszere **elküldi a bank felé** a vásárlási adatokat
e. Ha a **banki 3DS** ellenőrzés **sikertelen**, **és nincs jelen a vásárló** akkor a tranzakció itt sikertelenül ér véget, ugyanis ebben az esetben nem lehet challenge folyamatot indítani.
f. Ha a **banki 3DS** ellenőrzés **sikeres**, akkor a SimplePay rendszere beazonosítja a mentett kártyát, ami után megtörténik a fizetés banki hitelesítése (authorizáció).
g. A SimplePay rendszere szinkron választ ad a **do** hívásra az authorizáció eredményéről. Logikailag ez a pont megfelel a normál fizetés esetén a kereskedői weboldalra való visszairányítással (**back**).
h. Ezután a háttérben lefut a csalás megfigyelő és megelőző folyamat. Ha ennek során nem észlel a rendszer semmilyen problémát, akkor a háttérben visszajelzést küld a weboldal felé (**ipn**). Ez a fizetési tranzakció vége. Miután az IPN üzenet megérkezik, teljesítheti a megrendelést a kereskedő.

1.9 Letölthető segédletek

A sandbox és az éles rendszer kereskedői adminisztrációs felületéről egyformán letölthető a fejlesztéshez szükséges alap segédanyagok.

Mindkét rendszerben a bal oldali menüben a „**Letöltések**" menüpont alatt találhatók meg az alábbiak

- technikai dokumentáció
- mintakód
- logo csomag
- pénzügyi segédanyag

A fizetések dokumentációja és mintakódja az alábbi URL-en érhetők el:

Jelen dokumentáció magyarul:
http://simplepartner.hu/download.php?target=v21cardstoragedochu

Jelen dokumentáció angolul:
http://simplepartner.hu/download.php?target=v21cardstoragedocen

Oneclick és Recurring SDK mintakód:
http://simplepartner.hu/download.php?target=v21cardstoragesdk

Alap dokumentáció magyarul:
http://simplepartner.hu/download.php?target=v21dochu

Alap dokumentáció angolul:
http://simplepartner.hu/download.php?target=v21docen

Alap SDK mintakód:
http://simplepartner.hu/download.php?target=v21sdk

2 Implementáció oneclick tranzakciók esetén

2.1 start - oneclick kártyaregisztráció

A oneClick fizetés lehetőséget ad arra, hogy a vásárló a tranzakció során elmenthesse a bankkártyáját. A következő vásárlásnál már nem kell átirányítani a SimplePay fizető oldalára, hanem elegendő a háttérben elindítani a tranzakciót. Ennek előnye az, hogy a vásárló végig a kereskedő weboldalán marad és a fizetés átirányítás nélkül, egy háttérben történő hívás hatására történik meg.

Ezzel sokkal gyorsabb és kényelmesebb a fizetés a vásárló számára, ugyanis nem kell átirányítani a SimplePay fizető oldalára és nem kell minden vásárlásnál a kártyaadatot újra megadni.

A kártyáját elmentő vásárlónak a kereskedő rendszerében regisztrált felhasználónak kell legyen. Nem regisztrált felhasználó számára nem lehet ezt a szolgáltatást nyújtani!

A kereskedőnek a fizetés elindítása előtt lehetőséget kell biztosítson a vásárló számára, hogy egy egyedi azonosítót (**cardSecret, vagy kártyatitok**) adhasson meg.

A vásárlót tájékoztatnia kell a kereskedőnek arról, hogy a későbbiekben indítandó oneClick fizetéseknél csak a cardSecret vásárló általi megadásával tud majd a mentett kártyájával tranzakciót elindítani.

A cardSecret értékét a kereskedői rendszer nem mentheti, csak továbbíthatja a SimplePay felé.

A cardSecret a kártyaadat titkosított tárolásának is része. Mivel nincs önmagában tárolva a cardSecret értéke, így nincs lehetőség a módosításra sem. Abban az esetben, ha szükséges a változtatás, akkor törölni kell a regisztrált kártyát, majd egy új regisztrációt kell indítani.

A kártyaregisztrációval kapcsolatos adatok csak titkosított, SSL certifikációval rendelkező kapcsolaton keresztül történhetnek.

A regisztrációt indító kereskedői rendszernek TLS 1.2 https protokollt kell alkalmazzon. Önaláíró SSL certifikáció nem megfelelő!

A kártyaregisztrációs tranzakció elindítása egy olyan **start** hívás, ami ki van egészítve a cardSecret változóval. Minden egyéb szempontból (pl. **3DS**) azonos a korábban az alap dokumentációban leírt start funkcióval.

A regisztrációs tranzakció a továbbiakban nem különbözik a korábban leírt, a böngészőben történő normál fizetéstől.

A lenti JSON mintában pirossal lett jelölve a cardSecret. A további példákban végig a „**SECRET**" értéket használjuk a **cardSecret** tartalmaként.

```
{
  "merchant":"PUBLICTESTHUF",
  "orderRef":"101010515408918334286",
  "customer":"v2 START Tester",
  "customerEmail":"sdk_test@otpmobil.com",
  "language":"HU",
  "currency":"HUF",
  "total":"15",
  "sdkVersion":"SimplePay_PHP_SDK_2.0.2_181002",
  "salt":"56176471ee827d50540e5907dd99f979",
  "methods":[
    "CARD"
  ],
  "invoice":{
    "name":"SimplePay V2 Tester",
    "company":"",
    "country":"hu",
    "state":"Budapest",
    "city":"Budapest",
    "zip":"1111",
    "address":"Address 1",
    "address2":"",
    "phone":"06203164978"
  },
  "timeout":"2018-10-30T09:40:33+00:00",
  "url":"http://localhost/v21/back.php",
  "cardSecret":"SECRET"
}
```


A hívásra kapott válasz, a fizetési linkkel az alábbi:


```
{
  "salt":"gu3rMVuNGp5DMt5kXChmJJIQeKRZeEKA",
  "merchant":"PUBLICTESTHUF",
  "orderRef":"101010515408918334286",
  "currency":"HUF",
  "transactionId":99350378,
  "timeout":"2018-10-30T10:40:33+01:00",
  "total":15.0,
  "paymentUrl":"https://sandbox.simplepay.hu/pay/pay/pspHU/aoFtTKccntnBG6nfT2fORF2jd3iCAw1OwSn6A75b0jz9XtjzBp"
}
```


2.2 Oneclick kártya regisztrációs nyilatkozat

A kártyaregisztrációs tranzakció előtt a vásárlót tájékoztatni kell, ami során **a vásárlónak a kártya regisztrációról szóló nyilatkozatot** kifejezetten **el kell fogadja**. Ez a nyilatkozat nem helyettesíti az adattovábbítási nyilatkozatot, hanem azt kiegészíti a kártyaregisztrációs tranzakció esetén.

A nyilatkozat elhelyezésére több lehetőség is van
- az oldal saját Általános Szerződési Feltételeiben
- az oldal saját egyéb a fizetéssel kapcsolatos dokumentációjában
- a kártyaregisztrációs fizetésnél közvetlenül megjelenítve

A nyilatkozat elhelyezése a weboldalon önmagában nem elégséges, ha azzal a vásárló nem találkozik és nem fogadta el.

Az elfogadás történhet checkbox segítségével, vagy a tranzakció indításánál egyértelműen jelezve, hogy a fizetést elindítva egyben elfogadja a nyilatkozatot.
A kiemelt részen valós kereskedői adatokkal kell feltölteni a nyilatkozatot a következő módon.

URL: a szerződésben megadott domain név, vagy applikáció esetén ezt kiegészítve az applikáció nevével.

2.2.1 Magyar nyelvű tájékoztató

Az ismétlődő bankkártyás fizetés (továbbiakban: „Ismétlődő fizetés") egy, a SimplePay által biztosított bankkártya elfogadáshoz tartozó funkció, mely azt jelenti, hogy a Vásárló által a regisztrációs tranzakció során megadott bankkártyaadatokkal a jövőben újabb fizetéseket lehet kezdeményezni a bankkártyaadatok újbóli megadása nélkül. Az ismétlődő fizetés ún. „eseti hozzájárulásos" típusa minden tranzakció esetében a Vevő eseti jóváhagyásával történik, tehát, Ön valamennyi jövőbeni fizetésnél jóvá kell, hogy hagyja a tranzakciót. A sikeres fizetés tényéről Ön minden esetben a hagyományos bankkártyás fizetéssel megegyező csatornákon keresztül értesítést kap.

Az Ismétlődő fizetés igénybevételéhez jelen nyilatkozat elfogadásával Ön hozzájárul, hogy a sikeres regisztrációs tranzakciót követően jelen webshopban ***(……[URL]……..)*** *Ön az itt található felhasználói fiókjából kezdeményezett későbbi fizetések a bankkártyaadatok újbóli megadása nélkül menjenek végbe.*

Figyelem(!): a bankkártyaadatok kezelése a kártyatársasági szabályoknak megfelelően történik. A bankkártyaadatokhoz sem a Kereskedő, sem a SimplePay nem fér hozzá. A Kereskedő által tévesen vagy jogtalanul kezdeményezett ismétlődő fizetéses tranzakciókért közvetlenül a Kereskedő felel, Kereskedő fizetési szolgáltatójával (SimplePay) szemben bármilyen igényérvényesítés kizárt.

Jelen tájékoztatót átolvastam, annak tartalmát tudomásul veszem és elfogadom.

2.2.2 Angol nyelvű tájékoztató

Recurring credit card payment (hereinafter referred to as Recurring payment) is a function included in the acceptance of credit cards provided by SimplePay meaning that in the future it is possible to make payments with credit card details provided by the Customer during the registration transaction without giving credit card details again. One of the types of recurring payment, the so-called "ad-hoc payment" is made with the approval of the Customer during each transaction, so you have to approve all transactions in the case of each future payment. You will receive a notification of the successful payment through channels identical with conventional credit card payments in every case.

*By accepting this statement to use recurring payment, you allow to make subsequent payments made from your user account in this online store (****……[URL]……..****) without providing credit card details again after the successful registration.*

Please note: the processing of credit card details is in accordance with the rules of card issuers. Neither the merchant nor SimplePay has access to the credit card data. The Merchant shall assume direct liability for false or unauthorized recurring payments initiated by the Merchant, any claim enforcement against the Merchant's payment service provider (SimplePay) shall be unavailable.

I have read this notification, I take notice of its content and accept it.

2.3 Fizető oldal kártyaregisztráció esetén

A fizetőoldal csak egy ponton különbözik a normál fizetésnél látottaktól: a fizetést elindító gomb szövege ebben az esetben „KÁRTYAREGISZTRÁCIÓS FIZETÉS" lesz.

2.4 ipn – kártyaregisztráció esetén

Az IPN fogadása megegyezik a korábban leírt IPN fogadással, mivel az IPN független attól, hogy milyen módon lett elindítva a tranzakció.

A kártyatárolás esetén azonban az üzenet ki van bővítve az alábbi mezőkkel:

- **expiry**: a regisztrált kártya lejárati ideje
- **cardMask**: a kimaszkolt kártyaszám, ahol az utolsó 4 számjegy olvasható


```
{
  "cardMask":"xxxx-xxxx-xxxx-4193",
  "salt":"ywj9Sn8KLKUct08EApxxQfKWgCSlZZxm",
  "orderRef":"101010515606836609132",
  "method":"CARD",
  "merchant":" PUBLICTESTHUF",
  "finishDate":"2019-06-16T13:15:05+02:00",
  "expiry":"2021-10-31T00:00:00+02:00",
  "paymentDate":"2019-06-16T13:14:21+02:00",
  "transactionId":99204917,
  "status":"FINISHED"
}
```


Az IPN üzenetben annak a kártyának a lejárati ideje található meg, amivel a felhasználó végrehajtotta a kártyaregisztrációs fizetést. Ha ennek a kártyának rövidebb a hátralevő érvényessége, mint amit a kereskedő a token kérésnél beállított, akkor a tokenek is csak eddig az ideig lesznek aktívak, mert a kártya a lejárata után már nem lesz terhelhető.

2.5 do – tranzakció indítás mentett kártyával

A **do** hívást az API az alábbi URL-en várja:
https://sandbox.simplepay.hu/payment/v2/do

A „**do**" funkció használatával lehet fizetési tranzakciót indítani a korábban mentett kártya felhasználásával. Ebben az esetben nem kell a fizetőoldalra navigálni a vásárlót, mert a korábban megadott, a SimplePay rendszerében eltárolt kártyájával történik meg a fizetés. A POST metódussal küldött kérésre szinkron választ ad az API.

A fizetés elindítása előtt a kereskedő lehetőséget biztosít arra, hogy a weboldalon a vásárló a korábban a kártyájához regisztrált egyedi azonosítót (cardSecret) megadhassa. Ezzel az adattal, valamint a regisztrációs tranzakció SimplePay azonosítójával együtt kell elindítani a tranzakciót a SimplePay API felé.

A tárolt kártyás tranzakció indításának tehát az alábbi két kulcsfontosságú adat szükséges ahhoz, hogy a rendszer be tudja azonosítani a regisztrált kártyát.

Az egyik a korábbi, kártya regisztrációs tranzakciónak SimplePay azonosítója (**cardId**). Ez az adat a kereskedői rendszerben tárolható.

A másik szükséges adat a regisztrációs tranzakciónál a vásárló által megadott jelszó (**cardSecret**). Ezt az adatot a tranzakció indítása előtt kell bekérni és nem tárolható.

A cardSecret értékét a kereskedői rendszer nem mentheti, csak továbbíthatja a SimplePay felé.

2.6 do – 3DS ellenőrzéshez szükséges adatok

A **PSD2** megfelelőség miatt a felhasználó jelenlétével történő (**Customer Initiated Transaction**, vagy röviden **CIT)**, tárolt kártyás fizetések előtt is szükséges a kártyabirtokosi azonosítás, azaz a **3DS ellenőrzés**.

Ebben az esetben is küldenie kell a kereskedő rendszerének a korábban a „**start**" hívásban megadott, a **3DS** ellenőrzéshez szükséges adatokat.

A tranzakció indításhoz a fentieken kívül még három, a 3DS szempontjából fontos adatra van szükség:

Az egyik a tranzakció típusa (**type**) abból a szempontból, hogy a vásárló jelen van, vagy nincs a tranzakció indításakor. Oneclick esetén a „**do**" hívásban ez csak „**CIT**", azaz a vásárló jelenlétével elindított tranzakció lehet. Ez az adat a **3DS** ellenőrzéshez szükséges és a SimplePay rendszere továbbítja.

A másik adat a vásárló böngészőjének paraméterei (**browser**). Ez az adat is a **3DS** ellenőrzés része, ami továbbítva van a kártyakibocsátó bank felé.

browser tömb részletei

- **accept**: Accept http fejléc értéke
- **agent**: User-Agent http fejléc érték
- **ip**: a böngésző forrás IP-je
- **java**: a böngésző támogatja-e a java appleteket; javascriptben: *navigator.javaEnabled()*
- **lang**: a böngésző nyelve; javascriptben: *navigator.language*
- **color**: a böngésző színmélysége; javascriptben: *screen.colorDepth*
- **height** = a böngésző képernyőjének magassága; javascriptben: *screen.height*
- **width** = a böngésző képernyőjének szélessége; javascriptben: *screen.width*
- **tz** = a böngésző időzónája; javascriptben: *new Date().getTimezoneOffset()*

Mivel a kártyabirtokos jelen van a tranzakciónál (**CIT**) ezért szükséges a böngészőjének paramétereit átadni. A böngésző adatok hiánya esetén a banki 3DS ellenőrzés hibás eredménnyel zárulhat.

A harmadik a vásárló regisztrációjának módja (**threeDSReqAuthMethod**) a kereskedői rendszerben:
lehetséges értékek:

01: vendég
02: kereskedőnél regisztrált
05: harmadik feles azonosítóval regisztrált (Google, Facebook, account stb.)

Az adatokat a bank összehasonlíthatja adott bankkártya esetén a más online csatornákon ugyanarra a kártyára beérkező tranzakciók hasonló adataival. A valótlan adatok küldésének emiatt az a kockázta, hogy a 3DS ellenőrzés hibával zárul, azaz a fizetés meghiúsulhat.

A lenti mintakódban látható a tranzakció elindításához szükséges JSON string.

```
{
  "salt":"911f00255b2c080dd710fdb48586f9b4",
  "orderRef":"101010515409076376148",
  "cardId":"99350681",
  "cardSecret":"SECRET",
  "type": "CIT",
  "threeDSReqAuthMethod": "02",
  "browser": {
    "accept": "*/*",
    "agent": "Chrome\/75.0.3770.142 Safari\/537.36",
    "ip": "10.0.0.50",
    "java": false,
    "lang": "en",
    "color": 24,
    "height": 1080,
    "width": 1920,
    "tz": -120
  },
  "customerEmail":"sdk_test@otpmobil.com",
  "invoice":{
    "name":"SimplePay V2 Tester",
    "company":"",
    "country":"hu",
    "state":"Budapest",
    "city":"Budapest",
    "zip":"1111",
    "address":"Address 1",
    "address2":"",
    "phone":"06203164978"
  },
  "merchant":"PUBLICTESTHUF",
  "currency":"HUF",
  "customer":"DO tester",
  "total":42,
  "sdkVersion":"SimplePay_PHP_SDK_2.0.2_181002"
}
```

2.7 do – sikertelen 3DS ellenőrzéssel

Ha a **banki 3DS** ellenőrzés **sikertelen**, akkor az authorizáció még **nem** történik meg.

A kereskedői rendszernek fel kell készülnie arra, hogy a 3DS folyamat során a kártyakibocsátó bank megkövetelheti, hogy a kártyabirtokos interaktívan azonosítsa magát.

Ebben az esetben a „**do**” hívás válaszában a SimplePay rendszere visszaad egy olyan URL-t (**redirectUrl**) a kereskedőnek, amire átirányíthatja a vásárlót (**challenge**).

Válasz a „**do**” hívásra sikertelen 3DS ellenőrzés esetén

```
{
  "salt":"QnQtqSrVu01g331fNkXZBD2YusLzfG5U",
  "merchant":"PUBLICTESTHUF",
  "orderRef":"101010515409132276357",
  "currency":"HUF",
  "total":42.0
  "redirectUrl": "https://sandbox.simplepay.hu/pay/pay/ABCDEFG",
  "errorCodes": <errorCode>[]
}
```

2.8 do – challenge

Ha a kereskedői rendszer átirányítja a vásárlót a „**redirectUrl**” mezőben kapott URL-re, akkor egy olyan helyre jut, ahol a kártyakibocsátó bank által megkívánt további ügyfélellenőrzési folyamat történik meg.

Ha az ellenőrzés, majd a fizetés sikeres eredménnyel zárul, akkor a háttérben lefut a csalás megfigyelő és megelőző folyamat. Ha ennek során nem észlel a rendszer semmilyen problémát, akkor a háttérben visszajelzést küld a weboldal felé (**ipn**). Ez a fizetési tranzakció vége. Miután az IPN üzenet megérkezik, teljesítheti a megrendelést a kereskedő.

2.9 do – sikeres 3DS ellenőrzéssel

A sikeres banki 3DS ellenőrzés után a SimplePay rendszere beazonosítja a mentett kártyát, ami után megtörténik a fizetés banki hitelesítése (authorizáció). Sikeresség esetén tehát **nincs challenge**, így a fizetési folyamat a háttérben tud futni.

A hívásra a SimplePay rendszere szinkron választ ad az authorizáció eredményéről. Logikailag ez a pont megfelel a normál fizetés esetén a kereskedői weboldalra való visszairányítással (**back**).

Ezután a háttérben lefut a csalásfigyelő és megelőző folyamat. Ha ennek során nem észlel a rendszer semmilyen problémát, akkor a háttérben visszajelzést küld a weboldal felé (**ipn**). Ez a fizetési tranzakció vége. Miután az IPN üzenet megérkezik, teljesítheti a megrendelést a kereskedő.

A válaszban az általános adatokon kívül a **transactionId** tartalmazza a fizetés SimplePay tranzakció azonosítóját.

```
{
  "salt":"QnQtqSrVu01g331fNkXZBD2YusLzfG5U",
  "merchant":"PUBLICTESTHUF",
  "orderRef":"101010515409132276357",
  "currency":"HUF",
  "transactionId":99077246,
  "total":42.0
}
```

2.10 do – API v1 szolgáltatásban tárolt kártyák terhelése

Azon kereskedői rendszerek, ahol API v1-ről átváltanak API v2-re továbbra is tudják használni a korábban mentett kártyákat. A SimplePay szolgáltatásában korábban API v1 használatával elmentett kártyák terhelhetők API v2-vel is.

Mivel nem az API v2 dokumentációban leírt módon történt a kártyaregisztráció, hanem már a SimplePay rendszerében levő, korábban API v1-en regisztrált kártyát szeretnénk felhasználni, ezért a következő módon lehet ilyen esetben fizetést indítani a „**do**" végpont használatával:

- a SimplePay API v1 használatával elmentett azonosítót szükséges használni a mentett kártya beazonosításához (**IPN_CC_TOKEN**)
- a „do" indításakor **nem** kell elküldeni a „cardSecret" változót
- a „do" indításakor a type mezőben jelezni kell a felhasználó jelenlétét a „**type**" változóban
- a „do" indításakor a **„cardId"** változóban szükséges elküldeni a korábbi, OTP VPOS használatával elmentett kártya azonosítóját (**IPN_CC_TOKEN**)

Jelen dokumentáció későbbi részében tárgyalja a **legacy** kártyaregisztrációt és felhasználást mind oneclick, mind recurring folyamatokban. Az API v1 használatával tárolt és API v2 használatával terhelt kártyák esetén tulajdonképpen egy olyan legacy kártyaterhelés történik, ahol nem API v2-n történt a kártya mentése.
Ebből adódóan a teljes legacy folyamat megismeréséhez ajánlott elolvasni jelen dokumentáció „**Implementáció legacy kártyatárolás és tárolt kártyás fizetések esetén**" c. fejezetét.

Az alábbi példa mutatja erre az esetre a minimális mintakódot. Természetesen ezt ki kell egészíteni a „**do**" hívás leírásánál korábban részletezett adatokkal ahhoz, hogy a 3DS folyamat is megfelelően lefuthasson.

```
{
   "salt":"ed6f6279378476e1394ba3db85ad08e2",
   "orderRef":"101010515894412749644",
   "customerEmail":"sdk_test@otpmobil.com",
   "merchant":"PUBLICTESTHUF",
   "currency":"HUF",
   "cardId":"19123456",
   "type":"CIT",
   "threeDSReqAuthMethod": "02",
   "methods":[
      "CARD"
],
   "total":25,
   "sdkVersion":"SimplePay_PHP_SDK_2.0.9_200130:eef9c5a27ee155e74c65a536f194ff53"
}
```

2.10.1 Éles tranzakciók

Ez a megoldás az **éles** rendszeren az alábbi módon áll rendelkezésre:

- éles SimplePay API v1 használatával volt korábban elmentve a kártya
- élesített SimplePay fiókról van indítva a „**do**" hívás
- az éles SimplePay rendszerben tárolt regisztrációs ID van átadva a „**do**" hívás „**cardId**" mezőjében

A SimplePay rendszere lehetővé teszi azt, hogy az API v1 használatával elmentett kártyát **ugyanazon a kereskedői fiókon** API v2 alkalmazásával is terhelni lehessen.

Abban az esetben, ha külön fiókon szeretne indítani az API v1 és API v2 használatával a tranzakciókat, akkor létre kell hozni egy további éles fiókot. Ebben az esetben a SimplePay éles admin rendszerében össze kell kötni a kereskedő aktuális és az új SimplePay fiókját, hogy lehetőség legyen az átjárásra tárolt kártyák esetén a két fiók között.

Annak érdekében, hogy a kereskedői fiókok létrejöjjenek és megfelelően legyenek konfigurálva ez API v2 fejlesztés megkezdése előtt jelezze ezt sales kapcsolattartójának!

2.10.2 Teszt tranzakciók

A korábban éles rendszeren elmentett kártyák a sandbox rendszeren keresztül nem érhetők el!

A SimplePay **sandbox** rendszerben ez a szolgáltatás az alábbi módon tesztelhető:

- a sandbox SimplePay API v1 használatával volt korábban elmentve a kártya
- sandbox SimplePay fiókról van indítva a „**do**" hívás
- a sandbox SimplePay rendszerben tárolt regisztrációs ID van átadva a „**do**" hívás „**cardId**" mezőjében

A SimplePay rendszere lehetővé teszi azt, hogy az API v1 használatával elmentett kártyát **ugyanazon a kereskedői fiókon** API v2 alkalmazásával is terhelni lehessen.

Abban az esetben, ha külön fiókon szeretne indítani az API v1 és API v2 használatával teszt tranzakciókat, akkor létre kell hozni egy további sandbox fiókot. Ebben az esetben a SimplePay sandbox admin rendszerében össze kell kötni a kereskedő aktuális és az új SimplePay fiókját, hogy lehetőség legyen az átjárásra tárolt kártyák esetén a két fiók között.

Annak érdekében, hogy a sandbox kereskedői fiókok létrejöjjenek és megfelelően legyenek konfigurálva ez API v2 fejlesztés megkezdése előtt jelezze ezt sales kapcsolattartójának!

2.11 do – OTP Bank Webshop (Middleware, MW) szolgáltatásban tárolt kártyák terhelése

Azon kereskedői rendszerek, ahol OTP Webshop VPOS szolgáltatástól átváltanak SimplePay API v2-re továbbra is tudják használni a korábban mentett kártyákat. A SimplePay szolgáltatásában korábban OTP webshop használatával elmentett kártyák terhelhetők API v2-vel is.
Mivel nem az API v2 dokumentációban leírt módon történt a kártyaregisztráció, hanem már az OTP rendszerében levő, korábban regisztrált kártyát szeretnénk felhasználni, ezért a következő módon lehet ilyen esetben fizetést indítani a „**do**" végpont használatával:

- az OTP Webshop szolgáltatásában elmentett azonosítót szükséges használni a mentett kártya beazonosításához (**isConsumerRegistrationID**)
- a „do" indításakor **nem** kell elküldeni a „cardSecret" változót
a „do" indításakor a type mezőben jelezni kell a felhasználó jelenlétét a „**type**" változóban
- a „do" indításakor a **„cardId"** változóban szükséges elküldeni a korábbi, OTP VPOS használatával elmentett kártya azonosítóját (**isConsumerRegistrationID**)

Jelen dokumentáció későbbi részében tárgyalja a **legacy** kártyaregisztrációt és felhasználást mind oneclick, mind recurring folyamatokban. Az OTP Bank Webshop használatával tárolt és API v2 használatával terhelt kártyák esetén tulajdonképpen egy olyan legacy kártyaterhelés történik, ahol nem API v2-n történt a kártya mentése.
Ebből adódóan a teljes legacy folyamat megismeréséhez ajánlott elolvasni jelen dokumentáció „**Implementáció legacy kártyatárolás és tárolt kártyás fizetések esetén**" c. fejezetét.

Az alábbi példa mutatja erre az esetre a minimális mintakódot. Természetesen ezt ki kell egészíteni a „**do**" hívás leírásánál korábban részletezett adatokkal ahhoz, hogy a 3DS folyamat is megfelelően lefuthasson.

```
{
   "salt":"ed6f6279378476e1394ba3db85ad08e2",
   "orderRef":"101010515894412749644",
   "customerEmail":"sdk_test@otpmobil.com",
   "merchant":"PUBLICTESTHUF",
   "currency":"HUF",
   "cardId":"0123456789",
   "threeDSReqAuthMethod": "02",
   "methods":[
      "CARD"
],
   "total":25,
   "sdkVersion":"SimplePay_PHP_SDK_2.0.9_200130:eef9c5a27ee155e74c65a536f194ff53"
}
```

2.11.1 Éles tranzakciók

Ez a megoldás az **éles** rendszeren az alábbi módon áll rendelkezésre:
- éles OTP Webshop szolgáltatásban volt korábban elmentve a kártya
- élesített SimplePay fiókról van indítva a „**do**" hívás
- az éles OTP rendszerben tárolt regisztrációs ID van átadva a „**do**" hívás „**cardId**" mezőjében

A SimplePay éles admin rendszerében össze kell kötni a kereskedő aktuális SimplePay fiókját a kereskedő korábbi OTP VPOS szolgáltatásával. Ennek során válik aktívvá az átjárás a két szolgaáltatás között.

Annak érdekében, hogy az éles kereskedői fiókok megfelelően legyenek konfigurálva a szerződés megkötésekor jelezze ezirányú igényét!

2.11.2 Teszt tranzakciók

A korábban éles rendszeren elmentett kártyák a sandbox rendszeren keresztül nem érhetők el!

A SimplePay **sandbox** rendszerben ez a szolgáltatás az alábbi módon tesztelhető:

- a sandbox környezetben történik meg a kártyatárolás az OTP Webshop technikai megoldásával
- sandbox SimplePay fiókról van indítva a „**do**" hívás
- a sandbox rendszerben tárolt regisztrációs ID van átadva a „**do**" hívás „**cardId**" mezőjében

Ehhez a teszthez a SimplePay sandbox (teszt) rendszerében két kereskedői fiókot szükséges ehhez létrehozni:

egy fiókot, ami az OTP Webshop használatához van konfigurálva. Ezen a fiókon szükséges elvégezni a kártyaregisztrációs tranzakciót. Ennek kereskedő oldali leírása az alábbi dokumentációban található meg:

http://simplepartner.hu/download.php?target=otpvposhu

- **egy másik fiókot, ami az API v2 használatához van konfigurálva**. Ezen a fiókon szükséges indítani a „**do**" hívást a fentebb részletezett módon.

A SimplePay sandbox rendszerében össze kell kötni a kereskedő kétféle SimplePay fiókját. Ennek során válik aktívvá az átjárás a két szolgaáltatás között.

Annak érdekében, hogy a sandbox kereskedői fiókok megfelelően legyenek konfigurálva a szerződés megkötésekor jelezze ezirányú igényét!

2.12 cardquery – mentett kártyás tranzakciók lekérdezése

A **cardquery** hívást az API az alábbi URL-en várja:
https://sandbox.simplepay.hu/payment/v2/cardquery

A rögzített kártyával indított tranzakciók lekérdezhetők a SimplePay rendszerből. A lekérdezés lehetősége azután is megmarad, ha a vásárló törölte (inaktiválta) a kártyáját.

A lekérdezés indításához csak a kártya regisztrációs tranzakciónak SimplePay azonosítója (**cardId**) szükséges. Ez az adat a kereskedői rendszerben tárolható.

A lenti mintakód szemlélteti a tranzakció elindításához szükséges JSON stringet.

```
{
  "salt":"a97c24f6900754bdced622db157a1579",
  "cardId":"99077245",
  "merchant":"PUBLICTESTHUF",
  "history":false,
  "sdkVersion":"SimplePay_PHP_SDK_2.0.2_181002"
}
```

A válaszban az **expiry** mezőben található meg a regisztráció lejárati ideje, ami a kártya lejárat idejéből adódó maximális felhasználhatóság ideje.

A **status** mező tartalmazza a regisztráció érvényességének státuszát. Abban az esetben, ha az értéke „**ACTIVE**", akkor lehet további tranzakciókat indítani vele.
Ha ez az érték „**DISABLED**", akkor további tranzakció már nem indítható vele, úgy sem, ha a lejárati idő még nem érkezett el.


```
{
  "salt":"S6p1PJwd5pWvMALwhflIgQPSDgzpqG2a",
  "merchant":"PUBLICTESTHUF",
  "cardId":99077245,
  "status":"ACTIVE",
  "expiry":"2021-11-01T00:00:00+01:00"
}
```


Az adott kártyaregisztrációval végzett tranzakciók lekérdezéséhez használható a **history** paraméter **true** értékkel. Ha a „history" értéke **false**, vagy a paraméter nem szerepel a lekérdezés indításában, akkor csak a fenti, alap adatokat adja vissza az API a regisztrációról.


```
{
  "salt":"37026e8109998c70b6d550a3e9b1b083",
  "cardId":"99077245",
  "history":true,
  "merchant":"PUBLICTESTHUF",
  "sdkVersion":"SimplePay_PHP_SDK_2.0.2_181002"
}
```


A kibővített válaszban a fenti adatokon kívül megtalálhatók az adott regisztrált kártyával indított tranzakciók is.


```
{
  "salt":"MrAWnFQjxjPdGLeD8q6s96sbmjJuFxMs",
  "merchant":"PUBLICTESTHUF",
  "cardId":99077245,
  "status":"ACTIVE",
  "expiry":"2021-11-01T00:00:00+01:00",
  "history":[
    {
      "orderRef":"101010515409132276357",
      "transactionId":99077246,
      "status":"FINISHED",
      "paymentDate":"2018-10-30T16:27:08+01:00",
      "finishDate":"2018-10-30T16:27:23+01:00"
    },
    {
      "orderRef":"101010515409293187568",
      "transactionId":99077248,
      "status":"FINISHED",
      "paymentDate":"2018-10-30T20:55:19+01:00",
      "finishDate":"2018-10-30T20:55:31+01:00"
    }
  ]
}
```

2.13 cardcancel – mentett kártya törlése

A **cardcancel** hívást az API az alábbi URL-en várja:
https://sandbox.simplepay.hu/payment/v2/cardcancel

A rögzített kártya tulajdonosának biztosítani kell a lehetőséget a kártya törlésére abban az esetben, ha már nem szeretne további mentett kártyás vásárlásokat kezdeményezni az adott kereskedő weboldalán.

A kártya törlési lehetősége nem köthető feltételekhez. A kártya törlését a kereskedői weboldalon regisztrált felhasználó a saját profiljából el kell tudja indítani. A törlés módja egyértelmű és bármikor könnyen elérhető kell legyen a kártyatulajdonos számára.

A törlés lehetősége független attól, hogy milyen fizetések történtek már korábban, illetve attól is hogy milyen fizetéseknek kellene a későbbiekben megtörténni a regisztrált kártyával.

A kártya törlésének nem csak a kereskedői rendszerben kell megtörténjen, hanem a **cardcancel** használatával inaktiválni kell a SimplePay rendszerében is.

A tranzakciós adatok az inaktiválás után is lekérdezhetők lesznek, azonban a cardcancel hatására további tranzakció nem indítható már ezzel a regisztrációval
Az inaktiválás egyszeri és nem fordítható vissza. Ha ismét használni szeretné a vásárló az adott kártyát oneClick fizetésekhez, akkor ismét el kell mentse egy újabb kártyaregisztrációs fizetés során.

A törlés (inaktiválás) indításához csak a kártya regisztrációs tranzakciónak SimplePay azonosítója (**cardId**) szükséges. Ez az adat a kereskedői rendszerben tárolható.

A lenti mintakód szemlélteti a tranzakció elindításához szükséges JSON stringet.

```
{
  "salt":"e0592eaa4b704a85c4c3bbb4ee83323d",
  "cardId":"99077245",
  "merchant":" PUBLICTESTHUF",
  "sdkVersion":"SimplePay_PHP_SDK_2.0.2_181002"
}
```

Válaszban a status értéke „**DISABLED**” lesz a hívás hatására. Ezután nem lehet már újabb regisztrációs fizetést indítani az adott **cardId** használatával (**do**), azonban az adatok továbbra is lekérdezhetők maradnak (**cardquery**).

```
{
  "salt":"4CNlGqTIwmL4VQKZjrxu9Gp9VpfZ0XX0",
  "merchant":"PUBLICTESTHUF",
  "cardId":99077245,
  "status":"DISABLED",
  "expiry":"2021-11-01T00:00:00+01:00"
}
```

2.14 tokenquery - token állapotának lekérdezése

A **tokenquery** hívást az API az alábbi URL-en várja:
https://sandbox.simplepay.hu/payment/v2/tokenquery

Az API lehetőséget ad az egyedi token állapotának lekérdezésére.

A token aktívvá válik amikor a kártyaregisztrációs tranzakció során sikeres authorizáció történik.
A token akkor válik inaktívvá, amikor

- történik a felhasználásával egy sikeres tranzakció
- inaktiválja a kereskedő a tokent (**tokencancel** hívás)
- inaktiválja a kereskedő a kártyát (**cardcancel** hívás)
- lejár a kártya és automatikusan inaktívvá válik a SimplePay rendszerében

A hívásban az alábbi adatokat szükséges küldeni.
A „**token**" mezőben kell megadni azt a „**start**" hívás során generálódott konkrét tokent, aminek az állapotát szeretné lekérdezni.

```
{
   "token":"SPT2N8MLJ9P2VJ4FUC3X224UIED627RTQKORI3PMKPP6PVD7UEF99JF7NFNIGP5X",
   "merchant":"LRMIPSMHUF",
   "salt":"2480a8fb1efe5042d8d9ce8c22a957f4",
   "sdkVersion":"SimplePay_PHP_SDK_2.0.9_200130:3d383a4becf19088c05cc871fab5b1d7"
}
```

Válasz a tokenquery hívásra

```
{
   "salt":"pXCFBePh3JVLCcf5BFf9JfsbBn0mqFiK",
   "merchant":"LRMIPSMHUF",
   "token":"SPT2N8MLJ9P2VJ4FUC3X224UIED627RTQKORI3PMKPP6PVD7UEF99JF7NFNIGP5X",
   "status":"active",
   "expiry":"2020-12-01T17:00:00+01:00"
}
```

A válaszban a státusz mezőben található meg az adott token pillanatnyi állapota.
Lehetséges értékek:

- active: az adott tokennel lehetséges tranzakciót indítani
- used: az adott token már fel lett használva egy sikeres fizetéshez
- cancelled: az adott token inaktiválva lett (**tokencancel** hívással), vagy a regisztrációs fizetés

2.15 tokencancel - token inaktiválása

A **tokencancel** hívást az API az alábbi URL-en várja:
https://sandbox.simplepay.hu/payment/v2/tokencancel

Az API lehetőséget ad az egyedi token inaktiválására.

A hívásban az alábbi adatokat szükséges küldeni.
A „**token**" mezőben kell megadni azt a „**start**" hívás során generálódott konkrét tokent, amit inaktiválni szeretnénk.

```
{
   "token":"SPT2N8MLJ9P2VJ4FUC3X224UIED627RTQKORI3PMKPP6PVD7UEF99JF7NFNIGP5X",
   "merchant":"LRMIPSMHUF",
   "salt":"2480a8fb1efe5042d8d9ce8c22a957f4",
   "sdkVersion":"SimplePay_PHP_SDK_2.0.9_200130:3d383a4becf19088c05cc871fab5b1d7"
}
```


Válasz a tokencancel hívásra


```
{
   "salt":"DdH4qbkIcqpfCKsItpkGK5UfhE1sH9PX",
   "merchant":"LRMIPSMHUF",
   "token":"SPT2N8MLJ9P2VJ4FUC3X224UIED627RTQKORI3PMKPP6PVD7UEF99JF7NFNIGP5X",
   "status":"cancelled",
   "expiry":"2022-11-01T00:00:00+01:00"
}
```


A válaszban a státusz mezőben található meg az adott token pillanatnyi állapota, ami ebben az esetben cancelled.

3 PHP SDK használata oneclick tranzakciók esetén

A PHP SDK beállítása és használata az alap dokumentációban van részletesen tárgyalva. A kártyatárolási funkciók (**oneclick/recurring**) használatához az alábbi letölthető mintakód csomaggal kell kiegészíteni az alap SDK-t:
http://simplepartner.hu/download.php?target=v21cardstoragesdk

A kiegészítő csomag tartalmát a kicsomagolása után be kell másolni az SDK fő könyvtárába.

A csomag egyben tartalmazza a **oneclick** és **recurring** kártyatároláshoz szükséges elemeket is. Ezek közül az ebben a fejezetben tárgyalt **oneclick** esetén az alábbiak nem szükségesek. Jelenlétük nem okoz problémát, de akár törölhetők is.

- recurring mappa
- dorecurring.php
- src/SimplePayV21TokenData.php

Ezen kívül nem igényel semmilyen további beállítást az alapokhoz képest.

3.1 start – kártyatárolás

A **start** funkció használatával lehet a kártyaregisztrációs fizetést is indítani.

A start működési logikája kártyaregisztráció esetére a **start - oneclick kártyaregisztráció** fejezetben található meg. A továbbiakban a PHP SDK segítségével történő megvalósítását mutatja be ez a fejezet.

Mintakód az SDK-ban:

- **indexstorage.html** kártyatárolási mód kiválasztása
- **startstorage.php** kártyatárolás megvalósítása

Az oneclick kártyaregisztrációs tranzakció elindítása egy olyan **start** hívás, ami ki van egészítve a **2.1** fejezetben leírt **cardSecret** változóval. Minden egyéb szempontból azonos az alap dokumentációban leírt „start" funkcióval.

A mintakód egyben tartalmazza a oneclick és a recurring indításhoz szükséges mintakódot is.

A mintakódban oneclick esetére egy előre beállított értékkel (SECRET) van feltöltve a cardSecret változó. Ez ***helyett*** kell a vásárlótól bekérni ezt az adatot és a cardSecret-ben küldeni.

```
$trx->addData('cardSecret', 'SECRET');
```

3.2 ipn – kártyaregisztráció esetén

Az IPN fogadása megegyezik a korábban leírt IPN fogadással, mivel az IPN független attól, hogy milyen módon lett elindítva a tranzakció.

A kártyatárolás esetén azonban az üzenet ki van bővítve az alábbi mezőkkel:

- **expiry**: a regisztrált kártya lejárati dátuma
- **cardMask**: a kimaszkolt kártyaszám, ahol az utolsó 4 számjegy olvasható

3.3 do – fizetés tárolt kártyával

A **do** funkció használatával lehet a regisztrált kártyával **oneclick** tranzakciót indítani.

A do működési logikája a **do – tranzakció indítás mentett kártyával** fejezetben található meg. A továbbiakban a PHP SDK segítségével történő megvalósítását mutatja be ez a fejezet.

Mintakód az SDK-ban: **do.php**
A regisztrált kártyával indított tranzakciók esetén az alap SDK classokat tartalmazó fájlon kívül (src/SimplePayV21.php) szükséges még a src/SimplePayV21CardStorage.php fájl is. Ebben találhatók meg a kártyatároláshoz szükséges további funkciók.

```
//Import config data
require_once 'src/config.php';

//Import SimplePayment class
require_once 'src/SimplePayV21.php';

//Import SimplePayV21CardStorage class
require_once 'src/SimplePayV21CardStorage.php';

$trx = new SimplePayDo;

$trx->addConfig($config);

//account
$trx->addConfigData('merchantAccount', $_REQUEST['merchant']);

//OneClick transaction
$trx->addData('cardId', $_REQUEST['cardId']);

//cardSecret
$trx->addData('cardSecret', 'SECRET');

//threeDSReqAuthMethod
$trx->addData('threeDSReqAuthMethod', '02');

//add merchant transaction ID
$trx->addData('orderRef', '101010515537887919364');

// METHODS
$trx->addData('methods', array('CARD'));

//add currency
$trx->addData('currency', 'HUF');

//ORDER PRICE/TOTAL
$trx->addData('total', 42);
```

```
//customer's name
$trx->addData('customer', 'DO tester');

//customer's email
$trx->addData('customerEmail', 'sdk_test@otpmobil.com');

//start do
$trx->runDo();
```

Válasz a hívásra

```
Array
(
    [responseBody] => {"salt":"3odfGqZG5SbvvKSuWx0f9gggWpyajRYZ","merchant":"LRMIPSMHUF","orderRef":"10
10105155378879193 64","currency":"HUF","transactionId":99130180,"total":42.0}
    [responseSignature] => 2H9d7XPObSakFABMihoNmn08sSmty4HgoIxkXxu5Qu249lWoU3pmW9fwy7JjNd23
    [responseSignatureValid] => 1
    [salt] => 3odfGqZG5SbvvKSuWx0f9gggWpyajRYZ
    [merchant] => PUBLICTESTHUF
    [orderRef] => 101010515537887919364
    [currency] => HUF
    [transactionId] => 99130180
    [total] => 42
)
```

3.4 do – más szolgáltatásokban tárolt kártyák terhelése (API v1, vagy OTP Webshop)

Az API v2 **do** hívása alkalmas arra, hogy a korábban **API v1**, vagy az **OTP Bank Webshop VPOS** használatával tárolt kártyákat is lehessen terhelni.

A más rendszerekben elmentett kártyák terhelésével kapcsolatos részletes információ jelen dokumentáció alábbi fejezeteiben található meg:
API v1: do – API v2 fizetés API v1 használatával tárolt kártyával
OTP Webshop: do – API v2 fizetés OTP Webshop (Middleware, MW) szolgáltatásban tárolt kártyával

Az SDK-val indított **do** hívást ehhez az alábbiak szerint szükséges módosítani:

- a „do" indításakor **nem** kell elküldeni a „cardSecret" változót
- a „do" indításakor a **„cardId"** változóban szükséges elküldeni a korábbi API v1, vagy OTP Webshop VPOS használatával elmentett kártya azonosítóját

API v1 esetén az **IPN_CC_TOKEN** értékét kell a **cardId** mezőben küldeni.
OTP Webshop VPOS esetén az **isConsumerRegistrationID** értékét kell a **cardId** mezőben küldeni.

```
$trx->addData('cardId', 'abc123');
```

A **do** hívás egyéb paramétereit nem szükséges módosítani.

3.5 cardquery – tárolt kártya lekérdezése

A **cardquery** funkció használatával lehet a regisztrált kártyával indított tranzakciókat lekérdezni.

A cardquery működési logikája a **2.12** fejezetben található meg. A továbbiakban a PHP SDK segítségével történő megvalósítását mutatja be ez a fejezet.

Mintakód az SDK-ban: **cardquery.php**

```
//Import config data
require_once 'src/config.php';

//Import SimplePayment
require_once 'src/SimplePayV21.php';

//Import SimplePayV21CardStorage
require_once 'src/SimplePayV21CardStorage.php';

$trx = new SimplePayCardQuery;

//config
$trx->addConfig($config);

//account
$trx->addConfigData('merchantAccount', 'PUBLICTESTHUF');

//card ID
$trx->addData('cardId', '99123456');

$trx->runCardQuery();
```

Válasz a hívásra

```
Array
(
    [responseBody] => {"salt":"AtIpjA808vdA4kZadQFOXEvbOGAvdLtK","merchant":"LRMIPSMHUF","cardId":99130
179,"status":"ACTIVE","expiry":"2021-11-01T00:00:00+01:00"}
    [responseSignature] => xuMWvPKd5rj61hZYJ7MmkPb7/UYp2sqYlgi0iRl0g0j3ItskUYvPt+c+y1RHIojm
    [responseSignatureValid] => 1
    [salt] => AtIpjA808vdA4kZadQFOXEvbOGAvdLtK
    [merchant] => PUBLICTESTHUF
    [cardId] => 99130179
    [status] => ACTIVE
    [expiry] => 2021-11-01T00:00:00+01:00
)
```

A hívást ki lehet egészíteni a „**history**" változóval, ami eredményeképpen a tárolt kártyával végrehajtott fizetéseket lehet lekérdezni a rendszertől.

```
$trx->addData('history', true);
```

Ebben az esetben a válasz kibővül a „history" tömbbel, amiben megtalálhatók a tranzakciók.

```
Array
(
    [responseBody] => {"salt":"BsRupCoHXHSS0zCK11kFUVQ0SAOglXQx","merchant":"LRMIPSMHUF","cardId":99130
179,"status":"ACTIVE","expiry":"2021-11-01T00:00:00+01:00","history":[{"orderRef":"10101051553788791936
4","transactionId":99130180,"status":"FINISHED","paymentDate":"2019-03-28T16:59:52+01:00","finishDate":
"2019-03-28T17:00:03+01:00"}]}
    [responseSignature] => +Npu87rISjLb/UJFdLhEJhiyDEamkQxjonahHTL8XFI4mOEVKrM3a78viEqokHXv
    [responseSignatureValid] => 1
    [salt] => BsRupCoHXHSS0zCK11kFUVQ0SAOglXQx
    [merchant] => PUBLICTESTHUF
    [cardId] => 99130179
    [status] => ACTIVE
    [expiry] => 2021-11-01T00:00:00+01:00
    [history] => Array
        (
            [0] => Array
                (
                    [orderRef] => 101010515537887919364
                    [transactionId] => 99130180
                    [status] => FINISHED
                    [paymentDate] => 2019-03-28T16:59:52+01:00
                    [finishDate] => 2019-03-28T17:00:03+01:00
                )

        )

)
```

3.6 cardcancel – tárolt kártya törlése

A **cardcancel** funkció használatával lehet a regisztrált kártyát inaktiválni.

A cardcancel működési logikája a **2.13** fejezetben található meg. A továbbiakban a PHP SDK segítségével történő megvalósítását mutatja be ez a fejezet.
Mintakód az SDK-ban: **cardcancel.php**

```
//Import config data
require_once 'src/config.php';

//Import SimplePayment class
require_once 'src/SimplePayV21.php';

//Import SimplePayV21CardStorage
require_once 'src/SimplePayV21CardStorage.php';

$trx = new SimplePayCardCancel;

//config
$trx->addConfig($config);

//account
$trx->addConfigData('merchantAccount', 'PUBLICTESTHUF');

//add card ID
$trx->addData('cardId', '99123456');

//start cancel
$trx->runCardCancel();
```

Válasz a hívásra

```
Array
(
    [responseBody] => {"salt":"ATGG2rd6mZF0FzDpC0SG6lezldHEmUcS","merchant":" PUBLICTESTHUF ","cardId":
99123456,"status":"DISABLED","expiry":"2022-11-01T00:00:00+01:00"}
    [responseSignature] => JTYj5JWN8Ozmc/n61yqWGo4PSHcNrclsAxqLfuF/NG06DOAmMwQ0Sjqutx0sbGHA
    [responseSignatureValid] => 1
    [salt] => ATGG2rd6mZF0FzDpC0SG6lezldHEmUcS
    [merchant] => PUBLICTESTHUF
    [cardId] => 99123456
    [status] => DISABLED
    [expiry] => 2022-11-01T00:00:00+01:00
)
```

A válaszban a kártyaregisztrációnak DISABLED státuszba (pirossal jelölve) kellett kerülni a hívás hatására.

4 Implementáció recurring tranzakciók esetén

A tranzakciós adatokat a hivatkozott dokumentációban az "**Általános üzenetformátum**" fejezetben leírt módon egy JSON stringben szükséges küldeni, POST metódussal.

A hívások hitelesítése a hivatkozott dokumentációban az „**Üzenetek validálása**" fejezetben található meg.

4.1 start – recurring kártyaregisztrációval

Az ismétlődő fizetések esetén a kártya regisztrációja során a kereskedő rendszere meghatározott számú tokent kap a SimplePay rendszerétől. A későbbi fizetések során ezekkel a tokenekkel lehet elindítani a tranzakciókat.

Ezzel a fizetések automatizálhatók és a felhasználó jelenléte nélkül indíthatók.

A kártyáját elmentő vásárlónak a kereskedő rendszerében regisztrált felhasználónak kell legyen. Nem regisztrált felhasználó számára nem lehet ezt a szolgáltatást nyújtani!

A kereskedőnek a fizetés elindítása előtt tájékoztatnia kell a vásárlót arról, hogy a kártyája tárolva lesz. Továbbá tájékoztatnia kell a tárolt kártyához generálandó tokenek alábbi részleteiről is:

- hány darab token lesz létrehozva
- mekkora maximális összegű fizetésekhez lehet ezeket a továbbiakban felhasználni
- mi lesz a maximális lejárati ideje a tokeneknek, azaz meddig használhatók fel

A kártyaregisztrációval kapcsolatos adatok csak titkosított, SSL certifikációval rendelkező kapcsolaton keresztül történhetnek.

A regisztrációt indító kereskedői rendszernek TLS 1.2 https protokollt kell alkalmazzon. Önaláíró certifikáció nem megfelelő!

A kártyaregisztrációs tranzakció elindítása egy olyan **start** hívás, ami ki van egészítve a **recurring** változóval, ami a kért tokenek paramétereit (darabszám, maximális összeg, lejárat) tartalmazza az alábbiak szerint:

- **times**: hány alkalommal szeretné a kereskedő terhelni a vásárló kártyáját. Ennek megfelelő darabszámú token lesz generálva a hívás hatására
- **until**: a tokenek maximális érvényességének dátuma
- **maxAmount**: a tokenekkel indítható tranzakciók maximális összege

Ezen túlmenően a regisztrációs tranzakció minden szempontból (pl. **3DS**) azonos a korábban a hivatkozott dokumentációban leírt normál **start** funkcióval. A lenti JSON mintában pirossal van jelölve a recurring funkcióhoz tartozó többlet.

Hívás indítása


```
  "merchant":"PUBLICTESTHUF",
  "orderRef":"12700115510197967678",
  "customer":"v2 SimplePay Teszt",
  "customerEmail":"sdk_test@otpmobil.com",
  "language":"HU",
  "currency":"HUF",
  "sdkVersion":"SimplePayV2.1_PHP_SDK_2.0.4_180221",
  "salt":"c71e720d4a4f36ce6858cae698923cb1",
  "methods":[
    "CARD"
  ],
  "recurring":{
    "times":3,
    "until":"2020-12-01T18:00:00+02:00",
    "maxAmount":50
  },
  "threeDSReqAuthMethod": "02",
  "total":50,
  "invoice":{
    "name":"SimplePay V2 Tester",
    "company":"",
    "country":"hu",
    "state":"Budapest",
    "city":"Budapest",
    "zip":"1111",
    "address":"Address 1",
    "address2":"",
    "phone":"06203164978"
  },
  "timeout":"2019-02-24T15:59:56+01:00",
  "url":"http://localhost/v21/backrecurring.php",
}
```


Válasz a hívásra


```
{
  "salt":"jHseNFc16LUms0MFpHg0rkNQRgNNlbTd",
  "merchant":"PUBLICTESTHUF",
  "orderRef":"12700115510207958612",
  "currency":"HUF",
  "transactionId":99123339,
  "timeout":"2019-02-24T16:16:35+01:00",
  "total":25.0,
  "paymentUrl":"https://tsecurepay.simplepay.hu/pay/pay/pspHU/MyVrQUHeja9LQT6OwbuNItmTTpsbkQ4WwSNcMm8vA
X9KhbRmAH",
  "tokens":[
    "SPTB7NI5GE2WNM55DPKIA27H2QEVCVSVACQCS2VHCW9PVTTIWOVMWPLUSCAQ22SV",
    "SPTDAJMDGTOGMIAR6OGQU23IVBWFDCWL7XM542UUS2GJTH7K8CVQ6CE739BA763O",
    "SPT4CM2QVQ9GN2XNW2HJQ22TTONVDLIJVF4KO2RIQX94NJBR8GVLDO5526DA5E5F"
  ]
}
```


A válaszban az tokens mezőben találhatók meg a generált tokenek. A tokenek ezen a ponton még nem aktívak. A tokenek aktiválódásához a tranzakciót el kell indítani a korábban a start funkciónál leírt módon. Ha a tranzakció sikeres lesz, akkor az IPN üzenet megérkezésével azonos időben lesznek a tokenek is aktívak.

4.2 Kétlépcsős start – recurring kártyaregisztrációval

A kétlépcsős tranzakciók technikai engedélyezése a szerződés megkötésekor kérhető, illetve később is lehet jelezni a SimplePay IT support felé, ha erre a beállításra szükség van. Ennek megtörténte után a kereskedő rendszere tranzakciónként jelezheti, hogy egy-, vagy kétlépcsős fizetést szeretne indítani a **twoStep** változó értékével.

Abban az esetben, ha a tranzakció célja kizárólag a kártyaregisztráció, azaz a tranzakció összegét nem akarja a kereskedő terhelni, akkor kétlépcsős fiók esetén a folyamat egyszerűsíthető. Ebben az esetben a kereskedői rendszer a „**start**" hívásban az „**onlyCardReg**" változóval előre meghatározhatja, hogy a tranzakció célja kizárólag a kártyaregisztráció, azaz a tranzakció összegét nem akarja majd terhelni.

```
"onlyCardReg":true,
```

Kétlépcsős fizetés esetén ez annyit jelent, hogy a szükséges összeg sikeres befoglalása után a rendszer automatikusan elvégzi a teljes összegű feloldást is, így a kereskedői rendszernek nem szükséges egy újabb API hívásban elindítani a „**finish**" a hívást.

4.3 Recurring kártyaregisztrációs nyilatkozat

A kártyaregisztrációs tranzakció előtt a vásárlót tájékoztatni kell, ami során a vásárlónak a kártya regisztrációról szóló nyilatkozatot kifejezetten el kell fogadja. Ez a nyilatkozat nem helyettesíti az adattovábbítási nyilatkozatot, hanem azt kiegészíti a kártyaregisztrációs tranzakció esetén.

A nyilatkozat elhelyezésére több lehetőség is van

- az oldal saját Általános Szerződési Feltételeiben
- az oldal saját egyéb a fizetéssel kapcsolatos dokumentációjában
- a kártyaregisztrációs fizetésnél közvetlenül megjelenítve

A nyilatkozat elhelyezése a weboldalon önmagában nem elégséges, ha azzal a vásárló nem találkozik és nem fogadta el.

Az elfogadás történhet checkbox segítségével, vagy a tranzakció indításánál egyértelműen jelezve, hogy a fizetést elindítva egyben elfogadja a nyilatkozatot. A kiemelt részen valós kereskedői adatokkal kell feltölteni a nyilatkozatot a következő módon.

URL: a szerződésben megadott domain név, vagy applikáció esetén ezt kiegészítve az applikáció nevével.

4.3.1 Magyar nyelvű tájékoztató

Az ismétlődő bankkártyás fizetés (továbbiakban: „Ismétlődő fizetés") egy, a SimplePay által biztosított bankkártya elfogadáshoz tartozó funkció, mely azt jelenti, hogy a Vásárló által a regisztrációs tranzakció során megadott bankkártyaadatokkal a jövőben újabb fizetéseket lehet kezdeményezni a bankkártyaadatok újbóli megadása nélkül.

Az Ismétlődő fizetés igénybevételéhez jelen nyilatkozat elfogadásával Ön hozzájárul, hogy a sikeres regisztrációs tranzakciót követően jelen webshopban (……[URL]……..) kezdeményezett későbbi fizetések a bankkártyaadatok újbóli megadása és az Ön tranzakciónként hozzájárulása nélkül a Kereskedő által kezdeményezve történjenek.

Figyelem(!): a bankkártyaadatok kezelése a kártyatársasági szabályoknak megfelelően történik. A bankkártyaadatokhoz sem a Kereskedő, sem a SimplePay nem fér hozzá.

A Kereskedő által tévesen vagy jogtalanul kezdeményezett ismétlődő fizetéses tranzakciókért közvetlenül a Kereskedő felel, Kereskedő fizetési szolgáltatójával (SimplePay) szemben bármilyen igényérvényesítés kizárt.
Jelen tájékoztatót átolvastam, annak tartalmát tudomásul veszem és elfogadom.

4.3.2 Angol nyelvű tájékoztató

Recurring credit card payment (hereinafter referred to as Recurring payment) is a function included in the acceptance of credit cards provided by SimplePay meaning that in the future it is possible to make payments with credit card details provided by the Customer during the registration transaction without giving credit card details again.
By accepting this statement to use Recurring payment you allow to make subsequent payments made from your user account in this online store (……[URL]……..) without providing credit card details and you allow for the Merchant to make the payment without your transactional approval.

Please note: the processing of credit card details is in accordance with the rules of card issuers. Neither the merchant nor SimplePay has access to the credit card data.

The Merchant shall assume direct liability for false or unauthorized recurring payments initiated by the Merchant, any claim enforcement against the Merchant's payment service provider (SimplePay) shall be unavailable.
I have read this notification, I take notice of its content and accept it

4.4 Fizetőoldal

A fizetőoldal az alábbiakban különbözik a normál fizetéseknél látottaktól:

- a fizetést elindító gomb szövege ebben az esetben „KÁRTYAREGISZTRÁCIÓS FIZETÉS"
- a start kommunikáció során kért tokenek paraméterei (darabszám, maximális összeg és lejárati idő) meg vannak jelenítve

4.5 Back

Az authorizáció után a hivatkozott dokumentációban korábban leírt módon történik meg a kereskedői weboldalra a vásárló visszairányítása. A küldött paraméterek és feldolgozásuk recurring esetén nem különbözik a korábban leírtaktól.

4.6 ipn – kártyaregisztráció esetén

Az IPN fogadása megegyezik a korábban leírt IPN fogadással, mivel az IPN független attól, hogy milyen módon lett elindítva a tranzakció.

A kártyatárolás esetén azonban az üzenet ki van bővítve az alábbi mezőkkel:

- **expiry**: a regisztrált kártya lejárati ideje
- **cardMask**: a kimaszkolt kártyaszám, ahol az utolsó 4 számjegy olvasható


```
{
  "cardMask":"xxxx-xxxx-xxxx-4193",
  "salt":"ywj9Sn8KLKUct08EApxxQfKWgCS1ZZxm",
  "orderRef":"101010515606836609132",
  "method":"CARD",
  "merchant":" PUBLICTESTHUF",
  "finishDate":"2019-06-16T13:15:05+02:00",
  "expiry":"2021-10-31T00:00:00+02:00",
  "paymentDate":"2019-06-16T13:14:21+02:00",
  "transactionId":99204917,
  "status":"FINISHED"
}
```


Az IPN üzenetben annak a kártyának a lejárati ideje található meg, amivel a felhasználó végrehajtotta a kártyaregisztrációs fizetést. Ha ennek a kártyának rövidebb a hátralevő érvényessége, mint amit a kereskedő a token kérésnél beállított, akkor a tokenek is csak eddig az ideig lesznek aktívak, mert a kártya a lejárata után már nem lesz terhelhető.

4.7 dorecurring - fizetés indítása tokennel

A **dorecurring** hívást az API az alábbi URL-en várja:
https://sandbox.simplepay.hu/payment/v2/dorecurring

A „**dorecurring**" funkció használatával lehet fizetési tranzakciót indítani a korábban mentett kártya felhasználásával. Ebben az esetben nem kell a fizetőoldalra navigálni a vásárlót, mert a korábban megadott, a SimplePay rendszerében eltárolt kártyájával történik meg a fizetés. A POST metódussal küldött kérésre szinkron választ ad az API.

A kérés alapvetően megegyezik a korábban a „**oneclick**" fizetések esetén alkalmazott „**do**" hívással. Az egyetlen különbség, hogy a „**cardSecret**" változó helyett a „**token**" nevű változóban szükséges elküldeni az egyik korábban generált tokent.

4.8 dorecurring – 3DS ellenőrzéshez szükséges adatok

A **PSD2** megfelelőség miatt a **felhasználó jelenléte nélkül** történő, azaz a kereskedő által indított (**Merchant Initiated Transaction**) vagy röviden **MIT)**, tárolt kártyás fizetések előtt is szükséges a kártyabirtokosi azonosítás, azaz a **3DS ellenőrzés**.

A tranzakció indításakor szükséges küldeni a korábban generált tokenek (**token**) közül ez egyik olyat, ami még nem lett felhasználva

A tranzakció indításhoz a fentieken kívül még két, a **3DS** szempontjából fontos adatra van szükség.

Az egyik a tranzakció típusa (**type**) abból a szempontból fontos, hogy a vásárló jelen van, vagy nincs a tranzakció indításakor. Tokenes fizetés esetén a „**dorecurring**" hívásban ez „**MIT**" lehet mivel **a vásárló jelenléte nélkül** (recurring) lett elindítva a tranzakció
Ez az adat a **3DS** ellenőrzéshez szükséges és a SimplePay rendszere továbbítja.

A másik a vásárló regisztrációjának módja (**threeDSReqAuthMethod**) a kereskedői rendszerben: lehetséges értékek:

01: vendég
02: kereskedőnél regisztrált
05: harmadik feles azonosítóval regisztrált (Google, Facebook, account stb.)

Az adatokat a bank összehasonlíthatja adott bankkártya esetén a más online csatornákon ugyanarra a kártyára beérkező tranzakciók hasonló adataival. A valótlan adatok küldésének emiatt az a kockázta, hogy a 3DS ellenőrzés hibával zárul, azaz a fizetés meghiúsulhat.

Hívás indítása „**MIT**" esetén

```
{
  "salt":"776748178d05236f40839a4ecd0755dc",
  "orderRef":"101010515510244751578",
  "customerEmail":"sdk_test@otpmobil.com",
  "merchant":"PUBLICTESTHUF",
  "currency":"HUF",
  "customer":"Dorecurring tester",
  "token":"SPTB7NI5GE2WNM55DPKIA27H2QEVCVSVACQCS2VHCW9PVTTIWOVMWPLUSCAQ22SV",
  "type": "MIT",
  "threeDSReqAuthMethod": "02",
  "methods":[
    "CARD"
  ],
  "total":42,
  "invoice":{
    "name":"SimplePay V2 Tester",
    "company":"",
    "country":"hu",
    "state":"Budapest",
    "city":"Budapest",
    "zip":"1111",
    "address":"Address 1",
    "address2":"",
    "phone":"06203164978"
  },
  "sdkVersion":"SimplePay_PHP_SDK_2.0.4_180221:"
}
```

4.9 cardquery – tárolt kártyás tranzakciók lekérdezése

A **cardquery** hívás segítségével lehet a rendszerben rögzített kártyával indított tranzakciókat lekérdezni. Ennek módja a oneclick fizetés „cardquery – mentett kártyás tranzakciók lekérdezése" fejezetében leírtakkal megegyezik.

4.10 cardcancel – tárolt kártya törlése

A **cardcancel** hívás segítségével lehet a rendszerben rögzített kártyát törölni (inaktiválni). Ennek módja a oneclick fizetés „cardcancel – mentett kártya törlése" fejezetében leírtakkal megegyezik.

5 PHP SDK használata recurring tranzakciók esetén

A PHP SDK beállítása és használata az alap dokumentációban van részletesen tárgyalva. A kártyatárolási funkciók (**oneclick/recurring**) használatához az alábbi letölthető mintakód csomaggal kell kiegészíteni az alap SDK-t:
http://simplepartner.hu/download.php?target=v21cardstoragesdk

A kiegészítő csomag tartalmát a kicsomagolása után be kell másolni az SDK fő könyvtárába.

A csomag egyben tartalmazza a **oneclick** és **recurring** kártyatároláshoz szükséges elemeket is. Ezek közül az ebben a fejezetben tárgyalt **recurring** esetén az alábbi nem szükséges. Jelenléte nem okoz problémát, de akár törölhető is.

- do.php

A kiegészítő csomagban megtalálható „**recurring**" mappára írási engedélyt kell adni, mert ebben tárolja az SDK a létrehozott tokeneket.

Ezen kívül nem igényel semmilyen további beállítást az alap SDK-hoz képest.

A recurring mappa és azon belül a fájlban tárolt tokenek demonstrációs céllal vannak létrehozva. Ez a módszer kizárólag arra szolgál, hogy az SDK képes legyen szemléltetni a recurring működési logikáját.

Éles üzemmódban, valós recurring tranzakciók kezeléséhez ezen mód helyett az adatbázisban való tárolás ajánlott!

5.1 start

Mintakód az SDK-ban: **startstorage.php**
A hívás indítása az alábbi módon valósítható meg:

A mintakód egyben tartalmazza a oneclick és a recurring indításhoz szükséges mintakódot is.

A recurring funkció esetén a korábban tárgyalt **start** funkciót kell kiegészíteni a token generáláshoz szükséges a „recurring" elemekkel. Minden egyéb része a hívásnak megegyezik a korábban leírtaknak.

```
{
   "salt":"db35ed9444f04179b1e665db8552d858",
   "merchant":"LRMIPSMHUF",
   "orderRef":"101010515510319702312",
   "currency":"HUF",
   "sdkVersion":"SimplePay_PHP_SDK_2.0.9_200130:dc597528c6fb3b4c6312d470ed28895e",
   "methods":[
      "CARD"
   ],
   "recurring":{
      "times":3,
      "until":"2020-12-01T18:00:00+02:00",
      "maxAmount":50
   },
   "total":"25",
   "threeDSReqAuthMethod":"02",
```

```
   "customerEmail":"sdk_test@otpmobil.com",
   "language":"HU",
   "timeout":"2020-08-16T12:49:27+00:00",
   "url":"http:\/\/payusdk.simplelabs.hu\/nandi\/v21_200814\/backstorage.php",
   "invoice":{
      "name":"SimplePay V2 Tester",
      "country":"hu",
      "state":"Budapest",
      "city":"Budapest",
      "zip":"1111",
      "address":"Address 1",
      "address2":"",
      "phone":"06203164978"
   }
}
```


A hívásra kapott választ a $trx->transaction változóban található tömb tartalmazza. Ezen belül a „tokens" tömb tartalmazza a generált tokeneket a későbbi tranzakciókhoz.


```
Array
(
    [responseBody] => {"salt":"016u2eCj7v1nvP2wGGlwl7cP1n0SmRsy","merchant":"PUBLICTESTHUF","orderRef":
"101010515510319702312","currency":"HUF","transactionId":99123344,"timeout":"2019-02-24T19:22:50+01:00"
,"total":25.0,"paymentUrl":"https://tsecurepay.simplepay.hu/pay/pay/pspHU/ffWNRjXbQaZtxvZQnexbiFkzOFJnv
A8KxSJ1o1nfg0S6ejnDBG","tokens":["SPT9KF5FG8VQX944PXPM227HHFJ253O778EKS2XSQ6KX5GRQAKWELCTH9VOQXKML","SP
T96TF79WTQXG7FFN2O824RI5D25BPCWCSWK2RAJDEM72L3BGWCU6MXW35B36IU","SPT582Q3H7HQWI6GEFEW628LPVC247DAVL3QE2
S7UMC99WXW9CW84SUXLH7B5NHI"]}
    [responseSignature] => Pjeetn6DvKWLIPfq/nd30irXIIfWX7kHe3e4Fh61Ue4raXoS4dUyHT9ZwBg85GlT
    [responseSignatureValid] => 1
    [salt] => 016u2eCj7v1nvP2wGGlwl7cP1n0SmRsy
    [merchant] => PUBLICTESTHUF
    [orderRef] => 101010515510319702312
    [currency] => HUF
    [transactionId] => 99123344
    [timeout] => 2019-02-24T19:22:50+01:00
    [total] => 50
    [paymentUrl] => https://sandbox.simplepay.hu/pay/pay/pspHU/ffWNRjXbQaZtxvZQnexbiFkzOFJnvA8KxSJ1o1nf
g0S6ejnDBG
    [tokens] => Array
        (
            [0] => SPT9KF5FG8VQX944PXPM227HHFJ253O778EKS2XSQ6KX5GRQAKWELCTH9VOQXKML
            [1] => SPT96TF79WTQXG7FFN2O824RI5D25BPCWCSWK2RAJDEM72L3BGWCU6MXW35B36IU
            [2] => SPT582Q3H7HQWI6GEFEW628LPVC247DAVL3QE2S7UMC99WXW9CW84SUXLH7B5NHI
        )
)
```


5.2 back - tájékoztatások

A kereskedő oldalán a tájékoztatások megegyeznek a korábban leírt back tájékoztatásokkal. A kártyaregisztráció esetén nincs szükség itt többlet információ megjelenítésére.

5.3 ipn – kártyaregisztráció esetén

Az IPN fogadása megegyezik a korábban leírt IPN fogadással, mivel az IPN független attól, hogy milyen módon lett elindítva a tranzakció.

A kártyatárolás esetén azonban az üzenet ki van bővítve az alábbi mezőkkel:

- **expiry**: a regisztrált kártya lejárati dátuma
- **cardMask**: a kimaszkolt kártyaszám, ahol az utolsó 4 számjegy olvasható

5.4 dorecurring

Mintakód az SDK-ban: **dorecurring.php**
A hívás indítása az alábbi módon valósítható meg:

A dorecurring a korábban tárgyalt „**do**" hívás annyi különbséggel, hogy a „**cardSecret**" helyett a „**token**" változóban kell küldeni az egyik regisztrált token értékét. A hívásra az API szinkron választ ad.

Kérés

```
//Import config data
require_once 'src/config.php';
require_once 'src/SimplePayV21.php';

// new SimplePayDoRecurring instance
$trx = new SimplePayDoRecurring;

//add config data
$trx->addConfig($config);

//token
$trx->addData('token', 'SPT9KF5FG8VQX944PXPM227HHFJ253O778EKS2XSQ6KX5GRQAKWELCTH9VOQXKML');

//add merchant transaction ID
$trx->addData('orderRef', '101010515510244751578');

//threeDSReqAuthMethod
$trx->addData('threeDSReqAuthMethod', '02');

// METHODS
$trx->addData('methods', array('CARD'));

//add currency
$trx->addData('currency', 'HUF');

//ORDER PRICE/TOTAL
$trx->addData('total', 42);

//customer's name
$trx->addData('customer', 'Dorecurring tester');

//customer's email
$trx->addData('customerEmail', 'sdk_test@otpmobil.com');

//start dorecurring
$result = $trx->runDo();
```

Válasz

```
Array
(
    [responseBody] => {"salt":"j7Bge2EZoFH8ygegvdgz42ieEDa6LG4T","merchant":" PUBLICTESTHUF ","orderRef
":"101010515510244751578","currency":"HUF","transactionId":99172598,"total":42.0}
    [responseSignatureValid] => 1
    [salt] => j7Bge2EZoFH8ygegvdgz42ieEDa6LG4T
    [merchant] => PUBLICTESTHUF
    [orderRef] => 101010515510244751578
    [currency] => HUF
    [transactionId] => 99172598
    [total] => 42
)
```

5.5 cardquery

A cardquery hívás SDK segítségével történő megvalósítása megegyezik a korábban a oneclick kártyatárolásnál leírtakkal.

5.6 cardcancel

A cardcancel hívás SDK segítségével történő megvalósítása megegyezik a korábban a oneclick kártyatárolásnál leírtakkal.

6 Implementáció legacy kártyatárolás és tárolt kártyás fizetések esetén

Ezen a módon technikai szempontból egyszerűbbek lehetnek a regisztrációs és fizetési folyamatok, azonban nem a SimplePay saját rendszerében történik meg a kártya tárolása, hanem az OTP Bank rendszerében.

A tárolás helyének annyiban van jelentősége, hogy a SimplePay saját rendszerében tárolt kártyák esetén több elfogadó felé tudja indítani authorizációra a tranzakciókat attól függően, hogy adott pillanatban melyik érhető el. A banki rendszerben mentett kártyák esetén csak a Bank rendszerében történhet meg fizetéskor az authorizáció.

A kártyaregisztrációnál, vagy a kereskedői rendszerbe történő ügyfél regisztrációnál meg kell jeleníteni és el kell fogadtatnia kártyaregisztrációs nyilatkozatot.
A felhasználás módjától függően az ebben a dokumentációban megtalálható két nyilatkozat közül azt szükséges alkalmazni, ami lefedi a kereskedői folyamatot.

- oneclick kártyaregisztrációs nyilatkozat
- recurring kártyaregisztrációs nyilatkozat

A tranzakciós adatokat a hivatkozott dokumentációban az "**Általános üzenetformátum**" fejezetben leírt módon egy JSON stringben szükséges küldeni, POST metódussal.

A hívások hitelesítése a hivatkozott dokumentációban az „**Üzenetek validálása**" fejezetben található meg.

6.1 start – legacy oneclick kártyaregisztrációval

Legacy kártyaregisztráció során tárolt kártyák olyan fizetésekhez lesznek felhasználhatók, ahol a felhasználó akár jelen van, akár nincs, akkor is egy egyszeri fizetést indít el a kártyával a kereskedő.
A legacy recurringhez képest itt az egyszeri fizetésen van a hangsúly, míg a későbbiek során a recurringnél a rendszeres ismétlődés lesz a kulcs fogalom.

Az egyszeri fizetés megtörténhet a felhasználó jelenlétével és aktív közreműködésével, vagy a jelenléte nélkül is.

Amikor jelen van a vásárló (Customer Initiated Transaction, **CIT**),, akkor az eltárolt kártya egyfajta kényelmi szolgáltatásként alkalmazható, azaz a vásárló amikor a kereskedői oldalon megnyomja a fizetést elindító gombot, akkor a kereskedői rendszer a háttérben hívja meg a SimplePay rendszerét és ott történik meg a fizetési tranzakció.
Ennek előnye, hogy a vásárló nem lesz átirányítva a SimplePay fizetőoldalára és nem kell megadjon semmilyen további adatot, hanem egy kattintásra megtörténik a fizetés.

Amikor nincs jelen a vásárló (Merchant Initiated Transaction, **MIT**),, akkor az eltárolt kártyával a kereskedő tudja elindítani a fizetéseket. Ilyen lehet például az, amikor több mikro tranzakció történik, ahol nincs azonnal valós fizetés indítva, hanem a kereskedői rendszer naponta egyszer indítja el az aggregált összeggel a fizetést a SimplePay felé.

A kártyáját elmentő vásárlónak a kereskedő rendszerében regisztrált felhasználónak kell legyen. Nem regisztrált felhasználó számára nem lehet ezt a szolgáltatást nyújtani!

Ebben az esetben a start híváshoz hozzá kell fűzni a **legacyCardRegister** változót **true** értékkel.

Ezen túlmenően a regisztrációs tranzakció minden szempontból (pl. **3DS**) azonos a korábban a hivatkozott dokumentációban leírt normál **start** funkcióval. A fenti JSON mintában pirossal van jelölve a legacy oneclick funkcióhoz tartozó többlet.

Hívás indítása


```
  {
   "salt":"348e88ac5605ab6434421300e6a260f9",
   "merchant":"LRMIPSMHUF",
   "orderRef":"101010515975209557334",
   "currency":"HUF",
   "sdkVersion":"SimplePay_PHP_SDK_2.0.9_200130:dc597528c6fb3b4c6312d470ed28895e",
   "methods":[
      "CARD"
   ],
   "legacyCardRegister":true,
   "total":"25",
   "threeDSReqAuthMethod":"02",
   "customerEmail":"sdk_test@otpmobil.com",
   "language":"HU",
   "timeout":"2020-08-15T19:59:15+00:00",
   "url":"http:\/\/payusdk.simplelabs.hu\/nandi\/v21_200814\/backstorage.php",
   "invoice":{
      "name":"SimplePay V2 Tester",
      "country":"hu",
      "state":"Budapest",
      "city":"Budapest",
      "zip":"1111",
      "address":"Address 1",
      "address2":"",
      "phone":"06203164978"
   }
}
```


Válasz a hívásra


```
{
   "salt":"EULqEyvPShJfb7lfQFT1ykDZLtasktR7",
   "merchant":"LRMIPSMHUF",
   "orderRef":"101010515975209557334",
   "currency":"HUF",
   "transactionId":500068753,
   "timeout":"2020-08-15T21:59:15+02:00",
   "total":25.0,
   "paymentUrl":"https://sandbox.simplepay.hu/pay/pay/pspHU/P1tH6yhbj684031MsmGZVDmzWv07FAZRwShwJrOe8BeDbygmCo"
}
```


A válaszban visszakapott „**transactionId**" a létrejött tranzakció azonosítója. Ez lesz a későbbiekben az eltárolt kártyát azonosító egyedi adat, amit „**cardId**" néven kell a továbbiakban felhasználni a tárolt kártyás fizetések elindításánál.

A start hívásban létrejött tranzakció sikeres authorizációja után lesz aktív a mentett kártya további fizetések végrehajtásához.

6.2 do – legacy oneclick fizetés

A legacy oneclick céljára mentett kártya a „**do**" hívással terhelhető. A hívás korábban a „**do – tranzakció indítás mentett kártyával**" fejezetben lett részletesen tárgyalva.

A korábban leírt „**do**" híváshoz képest az egyetlen különbség az, hogy **nem szükséges küldeni** a **cardSecret** változót, hiszen az a **start** hívásban sem szerepelt.

Ha a vásárló jelen van akkor az alábbi módon történik a tárolt kártyás tranzakció indítása.
Ebben az esetben a **type** értéke **CIT** lesz és a **browser** adatok küldése is szükséges.
A korábban mentett kártyára a **cardId** mezőben megadott regisztrációs tranzakció azonosító mutat.

Mivel a vásárló jelen van, így értelmezhető a **challenge** folyamat is. A részletes leírása korábban a „do" kommunikációval összfüggésben az alábbi fejezetekben volt részletezve

- „**do – sikertelen 3DS ellenőrzéssel**"
- „**do – challenge**"

A **challenge** lehetősége miatt ebben az esetben is küldhető az „**url**" mező.


```
{
   "salt":"101f2e79fa37ecf489acbb13245f1f19",
   "orderRef":"101010515975236696434",
   "customerEmail":"sdk_test@otpmobil.com",
   "merchant":"LRMIPSMHUF",
   "currency":"HUF",
   "methods":[
      "CARD"
   ],
   "total":5,
   "cardId":"500068760",
   "threeDSReqAuthMethod":"02",
   "type":"CIT",
   "browser":{
      "accept":"text\/html,application\/xhtml+xml,application\/xml;q=0.9,image\/webp,image\/apng,*\/*;q
=0.8,application\/signed-exchange;v=b3;q=0.9",
      "agent":"Mozilla\/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit\/537.36 (KHTML, like Gecko) Chrom
e\/84.0.4147.105 Safari\/537.36",
      "ip":"94.199.53.96",
      "java":false,
      "lang":"hu-HU",
      "color":24,
      "height":1280,
      "width":720,
      "tz":-120
   },
   "url":"http:\/\/payusdk.simplelabs.hu\/nandi\/v21_200814\/back.php",
   "sdkVersion":"SimplePay_PHP_SDK_2.0.9_200130:dc597528c6fb3b4c6312d470ed28895e"
}
```

A hívásra kapott válasz megegyezik a „**do**" korábban leírt kimenetével.

```
{
   "salt":"anORX5M2d66mWskKWWcoqkqMtF58wYk7",
   "merchant":"LRMIPSMHUF",
   "orderRef":"101010515975236696434",
   "currency":"HUF",
   "transactionId":500068761,
   "total":5.0
}
```

Abban ez esetben, ha a kártyakibocsátó bank további interaktív ügyfélazonosítást kér a fizetésnél akkor a korábban a **challenge** folyamatnál leírt módon a „**do**" szinkron válaszában a „redirectUrl" mezőben adja vissza azt az URL-t a rendszer, ahova át kell irányítani a vásárlót.

```
{
  "salt":"QnQtqSrVu01g331fNkXZBD2YusLzfG5U",
  "merchant":"PUBLICTESTHUF",
  "orderRef":"101010515975236696434",
  "currency":"HUF",
  "total":5.0
  "redirectUrl": "https://sandbox.simplepay.hu/pay/pay/ABCDEFG",
  "errorCodes": <errorCode>[]
}
```

Ha a vásárló nincs jelen akkor az alábbi módon történik a tárolt kártyás tranzakció indítása. Ebben az esetben a **type** értéke **MIT** lesz. Mivel nincs jelen a vásárló és nem lehetséges az interaktív challenge folyamat, így nincs szükség a **browser** adatokra, valamint az **url** küldésére. A korábban mentett kártyára a **cardId** mezőben megadott regisztrációs tranzakció azonosító mutat.

```
{
   "salt":"101f2e79fa37ecf489acbb13245f1f19",
   "orderRef":"101010515975236696434",
   "customerEmail":"sdk_test@otpmobil.com",
   "merchant":"LRMIPSMHUF",
   "currency":"HUF",
   "methods":[
      "CARD"
   ],
   "total":5,
   "cardId":"500068760",
   "threeDSReqAuthMethod":"02",
   "type":"MIT",
   "sdkVersion":"SimplePay_PHP_SDK_2.0.9_200130:dc597528c6fb3b4c6312d470ed28895e"
}
```

A hívásra kapott válasz megegyezik a „**do**” korábban leírt kimenetével.


```
{
   "salt":"anORX5M2d66mWskKWWcoqkqMtF58wYk7",
   "merchant":"LRMIPSMHUF",
   "orderRef":"101010515975236696434",
   "currency":"HUF",
   "transactionId":500068761,
   "total":5.0
}
```


6.3 start – legacy recurring kártyaregisztrációval

Legacy recurring kártyaregisztráció során tárolt kártyák olyan fizetésekhez lesznek felhasználhatók, ahol a felhasználó nincs jelen és a kereskedő valamilyen rendszeres időintervallumban folyamatosan terheli a kártyát.
A hangsúly itt a rendszerességen, a megegyező időközökben történő terhelésen van. Az intervallumok lehetnek havi, heti, napi stb. a kereskedő üzleti modelljének megfelelően.

A kártyáját elmentő vásárlónak a kereskedő rendszerében regisztrált felhasználónak kell legyen. Nem regisztrált felhasználó számára nem lehet ezt a szolgáltatást nyújtani!

Ebben az esetben a start híváshoz hozzá kell fűzni a **legacyRecurring** változót **true** értékkel.

Ezen túlmenően a regisztrációs tranzakció minden szempontból (pl. **3DS**) azonos a korábban a hivatkozott dokumentációban leírt normál **start** funkcióval. A fenti JSON mintában pirossal van jelölve a legacy oneclick funkcióhoz tartozó többlet.

Hívás indítása


```
 {
  "salt":"348e88ac5605ab6434421300e6a260f9",
  "merchant":"LRMIPSMHUF",
  "orderRef":"101010515975209557334",
  "currency":"HUF",
  "sdkVersion":"SimplePay_PHP_SDK_2.0.9_200130:dc597528c6fb3b4c6312d470ed28895e",
  "methods":[
     "CARD"
  ],
  "legacyRecurring":true,
  "total":"25",
  "threeDSReqAuthMethod":"02",
  "customerEmail":"sdk_test@otpmobil.com",
  "language":"HU",
  "timeout":"2020-08-15T19:59:15+00:00",
  "url":"http:\/\/payusdk.simplelabs.hu\/nandi\/v21_200814\/backstorage.php",
  "invoice":{
     "name":"SimplePay V2 Tester",
     "country":"hu",
     "state":"Budapest",
     "city":"Budapest",
     "zip":"1111",
     "address":"Address 1",
     "address2":"",
     "phone":"06203164978"
```

```
    }
}
```

A válasz a hívásra

```
{
   "salt":"EULqEyvPShJfb7lfQFT1ykDZLtasktR7",
   "merchant":"LRMIPSMHUF",
   "orderRef":"101010515975209557334",
   "currency":"HUF",
   "transactionId":500068753,
   "timeout":"2020-08-15T21:59:15+02:00",
   "total":25.0,
   "paymentUrl":"https://sandbox.simplepay.hu/pay/pay/pspHU/P1tH6yhbj684031MsmGZVDmzWv07FAZRwShwJrOe8BeDbygmCo"
}
```

A válaszban visszakapott „**transactionId**" a létrejött tranzakció azonosítója. Ez lesz a későbbiekben az eltárolt kártyát azonosító egyedi adat, amit „**cardId**" néven kell a továbbiakban felhasználni a tárolt kártyás fizetések elindításánál.

A start hívásban létrejött tranzakció sikeres authorizációja után lesz aktív a mentett kártya további fizetések végrehajtásához.

6.4 do – legacy recurring fizetés

Mivel a vásárló nincs jelen és recurring fizetés fog történni, ezért ebben az esetben a **type** értéke **REC** lesz. Mivel nincs jelen a vásárló és nem lehetséges az interaktív challenge folyamat, így nincs szükség a **browser** adatokra, valamint az **url** küldésére.
A korábban mentett kártyára a **cardId** mezőben megadott regisztrációs tranzakció azonosító mutat.

Mivel a kártya recurring céljára lett elmentve, ezért a „do" hívásban is küldeni kell a **legacyRecurring** változót **true** értékkel.

```
{
   "salt":"101f2e79fa37ecf489acbb13245f1f19",
   "orderRef":"101010515975236696434",
   "customerEmail":"sdk_test@otpmobil.com",
   "merchant":"LRMIPSMHUF",
   "currency":"HUF",
   "methods":[
      "CARD"
   ],
   "total":5,
   "cardId":"500068760",
   "legacyRecurring":true,
   "threeDSReqAuthMethod":"02",
   "type":"REC",
   "sdkVersion":"SimplePay_PHP_SDK_2.0.9_200130:dc597528c6fb3b4c6312d470ed28895e"
}
```

A hívásra kapott válasz megegyezik a „**do**” korábban leírt kimenetével.


```
{
   "salt":"anORX5M2d66mWskKWWcoqkqMtF58wYk7",
   "merchant":"LRMIPSMHUF",
   "orderRef":"101010515975236696434",
   "currency":"HUF",
   "transactionId":500068761,
   "total":5.0
}
```

7 PHP SDK használata legacy kártyatárolás és tárolt kártyás tranzakciók esetén

A dokumentációhoz az SDK mintakódja is frissítve lesz. Annak alapján 2020.08.24-ig frissül ez a fejezet a mintakód felhasználásáról.

8 Logók és tájékoztatók

A regisztrációs fizetés esetén a fizetési elfogadóhely állandóan látható részén (pl. a láblécen), vagy a fizetés kiválasztásakor a tranzakciónál szükséges megjeleníteni a SimplePay logót.
A logót a hivatkozott dokumentáció „**Logók és tájékoztatók**" fejezetében leírtaknak megfelelően szükséges elhelyezni, megjeleníteni.

9 Adattovábbítási nyilatkozat

Az adattovábbítási nyilatkozat csak a kártyaregisztrációs tranzakció esetén szükséges.

Mivel a kereskedő harmadik félnek adja át a megrendelési/vásárlói adatokat, ezért a vásárlónak az adattovábbítási nyilatkozatot kifejezetten el kell fogadnia.
A nyilatkozatot a hivatkozott dokumentáció „**Adattovábbítási nyilatkozat**" fejezetében leírtaknak megfelelően szükséges megjeleníteni és elfogadtatni a vásárlóval.

Az adattovábbítási nyilatkozatot nem helyettesíti a korábban részletezett kártyatárolással kapcsolatos nyilatkozat.

10 Tesztelés

A SimplePay részéről minden élesítés előtt álló kereskedői rendszer (webáruház, mobil applikáció) tesztelésen esik át.

A kártyatárolás tesztelése a hivatkozott dokumentáció „**Tesztelés**" fejezetében leírtakat egészíti ki. Az alábbi pontokat abban az esetben is szükséges tesztelni, ha korábban élesített – és tesztelt - kereskedői rendszer továbbfejlesztéseként került implementálásra a kártyatárolási funkció.

10.1 Általános teszt pontok kártyatárolás esetére

Az alábbi teszt pontok minden kártyatárolási mód esetén érvényesek.

10.1.1 SSL certifikáció

- A kereskedői rendszer megfelelő SSL certifikációval rendelkezik

10.1.2 Regisztrált felhasználó

- A kártyáját csak a kereskedő weboldalának regisztrált felhasználója mentheti

10.1.3 Adattovábbítási nyilatkozat

- A szükséges nyilatkozat a leírtaknak megfelelően a regisztrációs tranzakció indítás előtt, vagy regisztrációkor megjelenik
- A nyilatkozatot, vagy az azt tartalmazó dokumentumot a vásárlónak kötelezően el kell fogadja
- A nyilatkozat a kereskedő adataival van feltöltve

10.1.4 Tájékoztatás a kártya regisztrációjáról

- A vásárló megfelelően tájékoztatva van arról, hogy a kártyája regisztrálva lesz a fizetési tranzakció során
- A vásárló tájékoztatva van arról, hogy miért lesz regisztrálva a kártyája és ez milyen fizetésekhez lesz használva a továbbiakban

10.1.5 Sikeres regisztrációs tranzakció

- A regisztrációs tranzakció megfelelően végig fut
- A szükséges tájékoztatások megjelennek a korábban leírtaknak megfelelően

10.1.6 Egynél több regisztráció

- Egynél több regisztráció esetén egymástól megkülönböztethetően meg vannak jelenítve a regisztrációk a vásárló számára.
- A vásárló kiválaszthatja, hogy a fizetéshez melyik regisztráció legyen használva.

10.1.7 Tárolt kártya törlése, inaktiválása

- A kereskedői rendszerből (weboldalról, mobil applikáció) indítva törölhető a mentett kártya a korábban leírtaknak megfelelően. A törlésnek (inaktiválásnak) a SimplePay rendszerében meg kell történjen (**cardcancel**).
- Egynél több regisztrálható kártya esetén a kártyatulajdonos kiválaszthatja, hogy melyik regisztrált kártyát kívánja törölni a kereskedői rendszerből

10.2 Teszt pontok oneclick kártyatárolás esetére

Az alábbi teszt pontok a fenti „**Általános teszt pontok kártyatárolás esetére**" fejezetben leírtakat egészítik ki **oneclick** fizetések esetén.

10.2.1 Tájékoztatás oneclick kártya regisztrációró esetén

- A vásárló tájékoztatva van arról, hogy a **cardSecret** mire szolgál a regisztráció és a későbbi fizetések során

10.2.2 Kártyatárolási nyilatkozat

- A szükséges **oneclick** kártyatárolási nyilatkozat a korábban leírtaknak megfelelően a tranzakció indítás előtt megjelenik és kötelezően el van fogadtatva.
- A nyilatkozat a korábban megjelenített adattovábbítási nyilatkozatot nem helyettesíti, hanem azt kiegészíti.

10.2.3 Oneclick tranzakció indítás regisztrált kártyával

- A kereskedői weboldalról a korábban leírtaknak megfelelően **cardSecret** használatával elindítható a fizetés a regisztrált kártyával (**do**).

10.3 Teszt pontok recurring kártyatárolás esetére

Az alábbi teszt pontok a fenti „**Általános teszt pontok kártyatárolás esetére**" fejezetben leírtakat egészítik ki **recurring** fizetések esetén.

10.3.1 Tájékoztatás recurring kártya regisztráció esetén

- A vásárló tájékoztatva van arról, hogy a tokenek milyen paraméterekkel lesznek generálva
    - hány darab
    - mekkora összegű
    - meddig érvényes

10.3.2 Kártyatárolási nyilatkozat

- A szükséges **recurring** kártyatárolási nyilatkozat a korábban leírtaknak megfelelően a tranzakció indítás előtt megjelenik és kötelezően el van fogadtatva.
- A nyilatkozat a korábban megjelenített adattovábbítási nyilatkozatot nem helyettesíti, hanem azt kiegészíti.

10.3.3 Recurring tranzakció indítás regisztrált kártyával

- **token** használatával elindítható a fizetés a regisztrált kártyával (**dorecurring**)

10.4 Teszt pontok legacy kártyatárolás esetére

Az alábbi teszt pontok a fenti „**Általános teszt pontok kártyatárolás esetére**" fejezetben leírtakat egészítik ki **legacy** fizetések esetén.

10.4.1 Tájékoztatás oneclick kártya regisztrációró esetén

- A vásárló tájékoztatva van arról, hogy a kártyája tárolva lesz további fizetésekhez

10.4.2 Kártyatárolási nyilatkozat

- A szükséges **oneclick** vagy **recurring** kártyatárolási nyilatkozat a korábban leírtaknak megfelelően a tranzakció indítás előtt megjelenik és kötelezően el van fogadtatva.

- A nyilatkozat a korábban megjelenített adattovábbítási nyilatkozatot nem helyettesíti, hanem azt kiegészíti.

10.4.3 legacy oneclick tranzakció indítás regisztrált kártyával

- ha a legacy fizetések **oneclick** logikában vannak implementálva, akkor a kereskedői weboldalról a korábban leírtaknak megfelelően elindítható a fizetés a regisztrált kártyával (**do**).

10.4.4 legacy recurring tranzakció indítás regisztrált kártyával

- ha a legacy fizetések **recurring** logikában vannak implementálva, akkor a kereskedői weboldalról a korábban leírtaknak megfelelően elindítható a fizetés a regisztrált kártyával (**do**).

11 Támogatás

További információért, technikai támogatásért kérjük lépjen kapcsolatba velünk az itsupport@otpmobil.com címen.
Kérjük, a hatékony ügyintézés végett minden esetben írja meg nekünk azt az adatot, ami alapján be tudjuk azonosítani a problémát, vagy a kérdését.

Tranzakció
Tranzakcióval kapcsolatos kérdés esetén a fizetés **SimplePay** azonosítóját adja meg nekünk. Az **azonosító nyolc számjegyű**, a sandbox esetében **1xxxxxxx**, éles esetén jelenleg **6xxxxxxx** formátumú.

Interface
A tranzakció a V1, vagy a V2 API használatával lett indítva.

Kereskedői fiók
Kereskedői technikai beállításokkal kapcsolatban a SimplePay rendszeren belüli **kereskedői fiók** azonosítót. Az azonosító a fiók **MERCHANT értéke**.

Fizetési rendszer
Melyik rendszerrel kapcsolatos a kérdése. A **sandbox rendszer** csak tesztek esetén, az **éles rendszer** a valós fizetési tranzakciók esetén.

Élesítés
Élesítési tesztek esetén kérjük, hogy írja meg nekünk az itsupport@otpmobil.com címre, hogy

- melyik szerződött domain névhez készült a tesztelhető fizetés
- melyik fiókot használják (MERCHANT)
- hol érhető el a tesztelhető rendszer