TAMGAES® V3.0

TamgaES Technology Development Studios

TamgaES.JS Dokümantasyonu

www.tamgaes.com

03.2025

TamgaES.JS – JavaScript Elektronik İmza Sağlayıcısı

TamgaES.JS Usb tokenlarda saklanan elektronik imza sertifikalarını kullanarak, kullanıcıya imza attırmak isteyen web tabanlı uygulamalar için geliştirilmiş, Only JS mantığı ile çalışan bir javascript kütüphanedir. Herhangi bir başka Javascript kütüphanesine bağımlılığı olmayıp pure js olarak geliştirilmiştir.

TamgaES.JS https://tamgaes.com/download adresinden indirilen TamgaES ile birlikte çalışır. TamgaES işletim sistemi bağımlılığı olmayan, kendi imza kütüphanesini kullanan çapraz platform (cross-platform application) uygulamadır.

TamgaES.JS ile entegre olmuş bir uygulama, TamgaES kurulu tüm sistemlerde her an takılı olan elektronik imzalara erişebilir, 5070 Sayılı imza kanunu ve ETSI Standartları çerçevesinde CAdES, PAdES ve XAdES imza yapılarının tüm varyasyonlarında imza oluşturabilir.

Kullanımı kısa ve basittir. 6 temel sınıf ve tanımlayıcı sınıflardan oluşur. TamgaESRequest, RequestParameters, TamgaESResponse, Smartcard, Certificate, TimeStamp.

1. Temel Sınıflar

TamgaESRequest	
İşlem isteği sınıfı. Bu sınıf ile akıllı kartlar, sertifikalar alınır, imza istekleri yapılır.	
Parameters	
requestParameters	RequestParameters sınıfı nesne.
Methods	
run()	İsteği gönderir. Sonuç message veya error eventlerine döner.
Events	
message	İstek başarılı olduğunda tetiklenir. Callback de TamgaESResponse sınıfı nesne döner.
error	Hata alınması durumunda tetiklenir. Callback de error code veya hata mesajı döner.

```
let tamgaESRequest = new TamgaESRequest(params);
```

RequestParameters		
İstek parametreleri sınıfı. TamgaESRequest sınıfının kurucu parametresidir.		
Properties		
request_type	Getter, Setter	İstek tipi. REQUEST_TYPE tanımlayıcı sınıfından bir değer alır.
smart_card_serial_number	Getter, Setter	E-İmza Token seri numarası.
certificate_serial_number	Getter, Setter	E-İmza Sertifika Seri Numarası.
pin	Getter, Setter	E-İmza Token Pin codu.
license_key	Getter, Setter	TamgaES lisans anahtarı.

valid_certificates_only	Getter, Setter	Sadece geçerli sertifikalar. Boolean tipi bir değer alır. Default: False
time_stamp	Getter, Setter	Zaman damgası bilgileri. TimeStamp sınıfı bir değer alır.
data_to_sign	Getter, Setter	İmzalanacak veri. Base64 tipi bir değer alır.
signature_type	Getter, Setter	İmza Türü. SIGNATURE_TYPE tanımlayıcı sınıfı bir değer alır. Default : SIGNATURE_TYPE.CAdES_BES
signature_profile	Getter, Setter	Türk İmza profilleri türü. SIGNATURE_PROFILE tanımlayıcı sınıfından bir değer alır. Default: SIGNATURE_PROFILE.None
signature_packaging	Getter, Setter	İmza paketleme türü. SIGNATURE_PACKAGING sınıfı bir değer alır. Default: SIGNATURE_PACKAGING.ATTACHED
signature_algorithm	Getter, Setter	İmzalama algoritması. SIGNATURE_ALGORITHMS sınıfı bir değer alır. Default: SIGNATURE_ALGORITHMS.RsaEncryption
pades_certification_level	Getter, Setter	PAdES imza seviye türü. PADES_CARTIFICATION_LEVEL türü bir değer alır. Default: PADES_CARTIFICATION_LEVEL.NOT_CERTIFIED
pades_reason	Getter, Setter	PAdES imza imzalama sebebi.
pades_contact	Getter, Setter	PAdES imza İletişim bilgisi.
pades_location	Getter, Setter	PAdES imza Lokasyon bilgisi.
pades_title	Getter, Setter	PAdES imza Unvan bilgisi.
pades_show_standart_sign_appearance	Getter, Setter	Standart PDF imza görüntüsünü aktif eder.
pades_standart_signappearance_X	Getter, Setter	Standart imza görüntüsü X koordinatı
pades_standart_sign_appearance_Y	Getter, Setter	Standart imza görüntüsü Y koordinatı
pades_standart_sign_appearance_opacity	Getter, Setter	Standart imza opaklık değeri

pades_addition_before_signature	Getter, Setter	PDF üzerine imza öncesi özelleştirilebilir metin veya resim ekler. JSON formatında değer alır. Bknz.
		
		[{
		"Text": "Örnek Metin 1",
		"Fontfilename": "Verdana.ttf",
		"FontSize": 14.0,
		"FontColorR": 0,
		"FontColorG": 0,
		"FontColorB": 255,
		"AddTo": 1,
		"LocationLeft": 10.0,
		"LocationBottom": 400.0,
		"Opacity": 80},
		{
		"Base64ImageData": "iVBORw0KGgoA...",
		"ImageWidth": 200,
		"ImageHeight": 80,
		"AddTo": 2,
		"LocationLeft": 10.0,
		"LocationBottom": 500.0,
		"Opacity": 50
		}]

```
let params = new RequestParameters();
params.license_key = $licenseKey;
params.request_type = REQUEST_TYPE.REQUEST_SIGN
params.smart_card_serial_number = $("#smartcards").children("option:selected").val();
params.certificate_serial_number = $("#certificates").children("option:selected").val();
params.pin = $('#pin').val();
params.data_to_sign = btoa($('#datatoSign').val());
```

TamgaESResponse

İşlem isteği yanıtıdır. TamgaESRequest sınıfı message event parametresidir.

Properties

smartcards	Getter	Smartcard sınıfı Dizi.
certificates	Getter	Certificate sınıfı Dizi
errorcode	Getter	Hata kodu. TAMGAES_ERRORS tanımlayıcı sınıfından bir değer alır.
state	Getter	Başarılı sonuç da “SUCCESS” değeri alır. Hata olması durumunda “ERROR” değeri alır.
warnings	Getter	Sistem uyarıları String dizi. Her yanıla birlikte gelir. Güncelleme gereksinimleri, sertifika bitişi uyarıları vs. son kullanıcıya gösterilmek üzere döndürülür.
signed_base64	Getter	Base64 tipi imzalanan veri.

```
tamgaESRequest.on("message", function (tamgaESResponse) {

    for (let i = 0; i < tamgaESResponse.smartcards.length; i++) {

        $('#smartcards').append($('<option/>', {
            value: tamgaESResponse.smartcards[i].serial_number,
            text: tamgaESResponse.smartcards[i].token_description
        }));
    }
});
```

Smartcard

Akıllı Kart bilgileri sınıfıdır. Doğrudan kullanıma kapalıdır. TamgaESResponse smartcards property sinin dönüş değeridir.

Properties

serial_number	Getter	Akıllı kart seri numarası.
token_description	Getter	Akıllı kart tanımlayıcı metin. Listelerde gösterim için kullanılır.
certificate	Getter	Akıllı karttaki sertifika

```
tamgaESRequest.on("message", function (tamgaESResponse) {

    for (let i = 0; i < tamgaESResponse.smartcards.length; i++) {

        $('#smartcards').append($('<option/>', {

            value: tamgaESResponse.smartcards[i].serial_number,
            text: tamgaESResponse.smartcards[i].token_description

        }));
    }
});
```

Certificate

İmza Sertifikası bilgileri sınıfıdır. Doğrudan kullanıma kapalıdır. TamgaESResponse sınıfının certificates property sinin veya Smartcard sınıfının certificates property sinin dönüş değeridir.

Properties

serial_number	Getter	İmza Sertifikası seri numarası
common_name	Getter	İmza Sertifikası sahibi adı.
identity	Getter	Akıllı karttaki sertifika sahibi tanımlayıcı tekil bilgi. Örn. Vatandaşlık No
country_name	Getter	Ülke adı veya kısaltması. Örn. Tr
start_date	Getter	Sertifika geçerlilik Başlangıç Tarihi
end_date	Getter	Sertifika geçerlilik Bitiş Tarihi
issuer_common_Name	Getter	Sertifikayı veren üretici tanımlayıcı bilgi.
is_valid_now	Getter	Sertifika şu an geçerli mi. True veya False döner.

```
tamgaESRequest.on("message", function (tamgaESResponse) {

        for (let i = 0; i < tamgaESResponse.certificates.length; i++) {

            $('#certificates').append($('<option/>', {
                value: tamgaESResponse.certificates[i].serial_number,
                text: tamgaESResponse.certificates[i].common_name
            }));

        }
});
```

TimeStamp

İmza da kullanılacak zaman damgası bilgileri sınıfıdır. RequestParameters sınıfının time_stamp parametresi setter değeridir.

Parameters

url	Zaman damgası erişim adresi
loginid	Zaman damgası kullanıcı adı bilgisi
password	Zaman damgası parolası
algorithm	Zaman damgası algoritması. ALGORITHMS tanımlayıcı sınıfından bir değer alır. Default: SHA256

```
let params = new RequestParameters();

params.time_stamp = new TimeStamp("http://test.timestamp.com", "12345, "345433",
ALGORITHMS.SHA256);
```

pades_addition_before_signature (İmza Öncesi Metin ve Resim)

PAdES imzaya özgü imza öncesi PDF üzerine metin ve görüntü eklemek için ayarlanır. JSON yapısında String bilgidir. RequestParameters sınıfının pades_addition_before_signature parametresi setter değeridir.

Properties

Text	Pdf üzerine eklenecek metin.
Base64ImageData	Pdf Üzerine eklenecek görüntünün base64 kodlanmış verisi
Fontfilename	Text özelliğindeki metne uygulanacak font. Font dosyası (Örn. Verdana.ttf)
FontSize	Text özelliğindeki metnin font boyutu.
FontColorR	Text özelliğindeki metnin font renginin RGB deki R değeri. 0-255 arasında bir değer.
FontColorG	Text özelliğindeki metnin font renginin RGB deki G değeri. 0-255 arasında bir değer.
FontColorB	Text özelliğindeki metnin font renginin RGB deki B değeri. 0-255 arasında bir değer.
AddTo	Metin veya görüntünün hangi sayfa[lara] ekleneceği bilgisi. **FIRST_PAGE** = 0, **LAST_PAGE** = 1, **ALL_PAGE** = 2
LocationLeft	Metin veya görüntünün PDF sayfasının sol kenarına olan uzaklığı.
LocationBottom	Metin veya görüntünün PDF sayfasının alt kenarına olan uzaklığı.
Opacity	Metin veya görüntünün opaklık değeri
ImageWidth	Base64ImageData özelliğindeki görüntünün genişliği
ImageHeight	Base64ImageData özelliğindeki görüntünün yüksekliği

Notlar;
Aşağıda iki elamandan oluşan bir JSON örneği verilmiştir. Bu örnek JSON imza öncesi PDF e bir "**İmzalayan Ad soyad**" ifadesini ve ikinci elemanda bulunan **Base64ImageData** görüntüyü ekler.

- Bir eleman ya **Text** ya da **Base64ImageData** içermelidir. Her ikisini de içermesi halinde Text veri dikkate alınır.
- **Ttf FontFilename TamgaESApp** imza uygulamanın kök dizininde **ExternalContent\fonts** klasöründe yer almalıdır. Dosyanın bulunamaması halinde varsayılan olarak Helvatica uygulanır.
- AddTo değeri enum value **(Örn. 0,1,2)** verilebileceği gibi enum name de kullanılabilir **(Örn. ALL_PAGE)**
- Metin içerisine yerleştirilebilecek tagler (çift köşeli parantez ile);

[[datetime]] : Günün tarihi
[[signername]] : İmza sertifikasındaki isim
[[location]] : RequestParameter nesnesinin ilgili özelliği
[[conctact]] : RequestParameter nesnesinin ilgili özelliği
[[reason]] : RequestParameter nesnesinin ilgili özelliği
[[title]] : RequestParameter nesnesinin ilgili özelliği

```
[{
  "Text": "İmzalayan : [[signername]]",
  "Fontfilename": "Verdana.ttf",
  "FontSize": 14.0,   "FontColorR": 0,   "FontColorG": 0,   "FontColorB": 255,
  "AddTo": 1,
  "LocationLeft": 10.0,   "LocationBottom": 400.0,
  "Opacity": 80},
{
  "Base64ImageData": "iVBORw0KGgoA...",
  "ImageWidth": 200,   "ImageHeight": 80,
  "AddTo": "ALL_PAGE",
  "LocationLeft": 10.0,   "LocationBottom": 500.0,
  "Opacity": 50
}]
```

2. Tanımlayıcı Sınıflar

REQUEST_TYPE	
İstek türü. RequestParemeters sınıfının request_type property setter değeridir.	
Enums	
REQUEST_SIGN	1000
REQUEST_SERIAL_SIGN	1001
REQUEST_PARALLEL_SIGN	1002
REQUEST_SMARTCARDS	1000011
REQUEST_CERTIFICATES_ON_SMARTCARD	1000012
REQUEST_SMARTCARDS_AND_CERTIFICATES	1000019

SIGNATURE_TYPE	
İmza türü. RequestParemeters sınıfının signature_type property setter değeridir.	
Enums	
NONE	0
CAdES_BES	1
CAdES_EPES	2
CAdES_T	3
CAdES_C	4
CAdES_XL	5
CAdES_XType1	6
CAdES_XType2	7
CAdES_XLType1	8
CAdES_XLType2	9
CAdES_A	10
PAdES_B	11
PAdES_T	12
PAdES_LT	13
PAdES_LTA	14

SIGNATURE_PACKAGING	
İmza paketleme türü. RequestParemeters sınıfının signature_packaging property setter değeridir.	
Enums	
ATTACHED	0 DEFAULT
DETACHED	1

SIGNATURE_PROFILE	
Türk imza profilleri türü. RequestParemeters sınıfının signature_profile property setter değeridir.	
Enums	
P1_1	"P1_1"
P2_1	"P2_1"
P3_1	"P3_1"
P4_1	"P4_1"
None	“None” DEFAULT

ALGORITHMS	
Zaman damgası algoritma türü. TimeStamp sınıfının kurucu parametresi algorithm değeridir.	
Enums	
SHA1	"SHA1"
SHA256	"SHA256" DEFAULT
SHA384	"SHA384"
SHA512	"SHA512"

PADES_CERTIFICATION_LEVEL	
PAdES imza imzalama seviyesidir. RequestParemeters sınıfının pades_certification_level property setter değeridir.	
Enums	
NOT_CERTIFIED	0 DEFAULT
CERTIFIED_NO_CHANGES_ALLOWED	1
CERTIFIED_FORM_FILLING	2
CERTIFIED_FORM_FILLING_AND_ANNOTATIONS	3

SIGNATURE_ALGORITHMS	
İmzalama Algoritması kodları. RequestParemeters sınıfının signature_algorithm property setter değeridir.	
Enums	
RsaEncryption	"RsaEncryption"
DsaWithSha224	"DsaWithSha224"
DsaWithSha256	"DsaWithSha256"
DsaWithSha384	"DsaWithSha384"
DsaWithSha512	"DsaWithSha512"
DsaWithSha1	"DsaWithSha1"
MD4WithRsa	"MD4WithRsa"
Md4WithRsaSignature	"Md4WithRsaSignature"

MD5WithRsa	"MD5WithRsa"
Sha1WithRsa	"Sha1WithRsa"
Md2WithRsaSignature	"Md2WithRsaSignature"
Md5WithRsaSignature	"Md5WithRsaSignature"
Sha1WithRsaSignature	"Sha1WithRsaSignature"
Sha224WithRsaSignature	"Sha224WithRsaSignature"
Sha256WithRsaSignature	"Sha256WithRsaSignature" DEFAULT
Sha384WithRsaSignature	"Sha384WithRsaSignature"
Sha512WithRsaSignature	"Sha512WithRsaSignature"
ECDsaWithSha1	"ECDsaWithSha1"
ECDsaWithSha224	"ECDsaWithSha224"
ECDsaWithSha256	"ECDsaWithSha256"
ECDsaWithSha384	"ECDsaWithSha384"
ECDsaWithSha512	"ECDsaWithSha512"
IdDsaWithSha1	"IdDsaWithSha1"
id_TA_ECDSA_SHA_1	"id_TA_ECDSA_SHA_1"
id_TA_ECDSA_SHA_224	"id_TA_ECDSA_SHA_224"
id_TA_ECDSA_SHA_256	"id_TA_ECDSA_SHA_256"
id_TA_ECDSA_SHA_384	"id_TA_ECDSA_SHA_384"
id_TA_ECDSA_SHA_512	"id_TA_ECDSA_SHA_512"
id_TA_RSA_v1_5_SHA_1	"id_TA_RSA_v1_5_SHA_1"
id_TA_RSA_v1_5_SHA_256	"id_TA_RSA_v1_5_SHA_256"
id_TA_RSA_PSS_SHA_1	"id_TA_RSA_PSS_SHA_1"
id_TA_RSA_PSS_SHA_256	"id_TA_RSA_PSS_SHA_256"
GostR3411x94WithGostR3410x94	"GostR3411x94WithGostR3410x94"
GostR3411x94WithGostR3410x2001	"GostR3411x94WithGostR3410x2001"
id_tc26_signwithdigest_gost_3410_12_256	"id_tc26_signwithdigest_gost_3410_12_256"
id_tc26_signwithdigest_gost_3410_12_512	"id_tc26_signwithdigest_gost_3410_12_512"

TAMGAES_ERRORS

Hata kodları. TamgaESResponse sınıfının errorcode property getter değeridir.

Enums	
TAMGAES_IMZA_UYGULAMASINA_BAGLANILAMADI	1000
PIN_BLOKE	30000
UYGULAMA_IC_HATASI_OLUSTU	40000
BEKLENMEYEN_HATA_OLUSTU	50000
HATALI_PIN	50001
TAKILI_KART_BULUNAMADI	50002
KARTTA_SERTIFIKA_YOK	50003
SECILEN_KART_SISTEMDE_TAKILI_DEGIL	50004
DOSYA_VERISI_DONUSTURULEMEDI	50005
IMZALAMA_HATASI	50009

ISTEK_OKUNAMIYOR_VEYA_ISLENEMIYOR	50010
TALEP_TURU_TESPIT_EDILEMEDI	50011
ISTEMCIDE_TAMGAES_KURULU_DEGIL	50013
ZORUNLU_GUNCELLE_YAPILMASI_GEREKIYOR	50014
LISANS_GEREKIYOR	50018
GECERSIZ_LISANS_ANAHTARI	50019
LISANS_SURESI_BITMIS	50020
FONKSIYON_LISANS_YETKISINI_ASIYOR	50021
LISANS_HATASI	50022

3. **Sistemde takılı akıllı kartların listesini *#smartcard* id'li listeye doldur.**
   (REQUEST_SMARTCARDS)

```
//İstek parametreleri
let params = new RequestParameters();
params.license_key = licenseKey;
params.request_type = REQUEST_TYPE.REQUEST_SMARTCARDS;
let tamgaESRequest = new TamgaESRequest(params);

tamgaESRequest.on("message", function (tamgaESResponse) {

   if(tamgaESResponse.smartcards.length == 0) {
       alert("Takılı Kart Bulunamadı");
       return;
   }

   //Akıllı kartlarda dön ve listeye doldur.
   for (let i = 0; i < tamgaESResponse.smartcards.length; i++) {
       $('#smartcards').append($('<option/>', {
           value: tamgaESResponse.smartcards[i].serial_number,
           text: tamgaESResponse.smartcards[i].token_description
       }));
   }

   //Olası uyarıları yazdır.(Güncelleştirme var, sertifika bitiyor vs.)
   for (let i = 0; i < tamgaESResponse.warnings.length; i++) {
       console.warn("WARNING : " + e.warnings[0]);
   }

});

//Hata Event i
tamgaESRequest.on("error", function (tamgaes_error_code) {
   alert(tamgaes_error_code);
});

//isteği gönder
tamgaESRequest.run();
```

4. **Sistemde takılı akıllı kartlardan sertifikaları oku ve *#certificates* id'li listeye doldur.** (REQUEST_CERTIFICATES_ON_SMARTCARD)

```
//İstek parametrelerini hazırla
let params = new RequestParameters();
params.license_key = licenseKey;
params.request_type = REQUEST_TYPE.REQUEST_CERTIFICATES_ON_SMARTCARD;
params.smart_card_serial_number = card_serial_number;
let tamgaESRequest = new TamgaESRequest(params);

tamgaESRequest.on("message", function (tamgaESResponse) {

   if(tamgaESResponse.certificates.length == 0) {
       alert("Kartta sertifika yok ya da okunamadı");
       return;
   }


   //Serfikalar listesinde dön ve listeye doldur
   for (let i = 0; i < tamgaESResponse.certificates.length; i++) {

       $('#certificates').append($('<option/>', {
           value: tamgaESResponse.certificates[i].serial_number,
           text: tamgaESResponse.certificates[i].common_name
       }));
   }

   //Olası uyarıları yazdır.(Güncelleştirme var, sertifika bitiyor vs.)
   for (let i = 0; i < tamgaESResponse.warnings.length; i++) {
       console.warn("WARNING : " + e.warnings[0]);
   }

});

//Hata event i
tamgaESRequest.on("error", function (tamgaes_error_code) {
   alert(tamgaes_error_code);
});

//isteği gönder
tamgaESRequest.run();
```

5. **Sistemde takılı akıllı kart ve sertifikaları tek seferde çekme.**
(REQUEST_SMARTCARDS_AND_CERTIFICATES)

Bu istek yapıldığında akıllı kartlar çekilirken smartcard nesnesinin certifcate özelliği doldurulur.

```
let params = new RequestParameters();
params.license_key = licenseKey;
params.request_type = REQUEST_TYPE.REQUEST_SMARTCARDS_AND_CERTIFICATES;

let tamgaESRequest = new TamgaESRequest(params);

tamgaESRequest.on("message", function (tamgaESResponse) {

        if(tamgaESResponse.smartcards.length == 0) {
            alert("Takılı Kart Bulunamadı");
            return;
        }


        for (let i = 0; i < tamgaESResponse.smartcards.length; i++) {

              //Akıllı kart seri numarasını ve etiketini yazdır
              console.info(tamgaESResponse.smartcards[i].serial_number);
              console.info(tamgaESResponse.smartcards[i].token_description);

              //Akıllı karttaki sertifikanın seri numarasını ve adını yazdır
              console.info(tamgaESResponse.smartcards[i].certificates[0].serial_number;
              console.info(tamgaESResponse.smartcards[i].certificates[0].common_name;

        }

        //Olası uyarıları yazdır.(Güncelleştirme var, sertifika bitiyor vs.)
        for (let i = 0; i < tamgaESResponse.warnings.length; i++) {
            console.warn("WARNING : " + e.warnings[0]);
        }
});

tamgaESRequest.on("error", function (tamgaes_error_code) {
    console.error(tamgaes_error_code);
});

tamgaESRequest.run();
```

6. **Bir veriye CAdES imza atma / seri imza ekleme / paralel imza ekleme**
(REQUEST_SIGN / REQUEST_PARALLEL_SIGN / REQUEST_SERIAL_SIGN)

İsteğin REQUEST_PARALLEL_SIGN veya REQUEST_SERIAL_SIGN olarak gönderilmesi durumunda, RequestParameters in data_to_sign özelliği, zaten en az bir imza taşıyan CAdES formatında base64 kodlanmış dosya verisi olmalıdır. Örnekte Tümleşik CAdES-BES imza atılmıştır.

```
//İmza İsteği parametrelerini hazırla
let params = new RequestParameters();
params.license_key = licenseKey;
params.request_type = REQUEST_TYPE.REQUEST_SIGN
params.smart_card_serial_number=$("#smartcards").children("option:selected").val();
params.certificate_serial_number=$("#certificates").children("option:selected").val();
params.pin = $('#token_password').val();
params.data_to_sign = base64FileData;
params.signature_type = SIGNATURE_TYPE.CAdES_BES;
params.signature_algorithm = SIGNATURE_ALGORITHMS.Sha256WithRsaSignature;
params.signature_packaging = SIGNATURE_PACKAGING.ATTACHED;  // Tümleşik İmza
// params.time_stamp=new TimeStamp("http://test.ts.com","456","***",  ALGORITHMS.SHA256);
// params.signature_profile = SIGNATURE_PROFILE.P4_1;

let tamgaESRequest = new TamgaESRequest(params);

tamgaESRequest.on("message", function (tamgaESResponse) {

   //İmzalanan dosya base64 verisini yazdır
   console.info(e.signed_base64);

    //Uyarılar. Örn. CAdES-XL istenmesine rağmen zaman damgası parametresi boş  vb.
    for (let i = 0; i < tamgaESResponse.warnings.length; i++) {
        console.warn("WARNING : " + tamgaESResponse.warnings[0]);
    }

});


//Hata event i
tamgaESRequest.on("error", function (tamgaes_error_code) {
   alert(tamgaes_error_code);
});

//İsteği gönder
tamgaESRequest.run();
```

7. Bir veriye PAdES imza atma / paralel imza ekleme (REQUEST_SIGN / REQUEST_PARALLEL_SIGN)

İsteğin REQUEST_PARALLEL_SIGN olarak gönderilmesi durumunda, RequestParameters in data_to_sign özelliği, zaten en az bir imza taşıyan PAdES formatında base64 kodlanmış dosya verisi olmalıdır. Örnekte Tümleşik PAdES-LTA imza atılmıştır.

```
let params = new RequestParameters();
params.license_key = licenseKey;
params.request_type = REQUEST_TYPE.REQUEST_SIGN;
params.smart_card_serial_number = $("#smartcards").children("option:selected").val();
params.certificate_serial_number = $("#certificates").children("option:selected").val();
params.pin = $('#token_password').val();
params.data_to_sign = base64FileData;
params.signature_type = SIGNATURE_TYPE.PAdES_LTA;
params.signature_algorithm = SIGNATURE_ALGORITHMS.Sha256WithRsaSignature;
params.signature_packaging = SIGNATURE_PACKAGING.ATTACHED;  // Tümleşik İmza
params.time_stamp = new TimeStamp("http://test.ts.com","456","***",  ALGORITHMS.SHA256);
params.signature_profile = SIGNATURE_PROFILE.P4_1;
params.pades_location = "Lokasyon";
params.pades_contact = "İletişim";
params.pades_reason = "Sebep";
params.pades_title = "Unvan";
params.pades_certification_level = PADES_CARTIFICATION_LEVEL.NOT_CERTIFIED;
params.pades_show_standart_sign_appearance = true;
params.pades_standart_signappearance_X = 10;
params.pades_standart_sign_appearance_Y = 100;
params.pades_standart_sign_appearance_opacity = 40;
params.pades_addition_before_signature = '[' +
    '                                {' +
    '                                  "Base64ImageData": "iVBORw0KGgoAAAANSUh…' +
    '                                  "ImageWidth": 200,' +
    '                                  "ImageHeight": 80,' +
    '                                  "AddTo": 2,' +
    '                                  "LocationLeft": 10.0,' +
    '                                  "LocationBottom": 500.0,' +
    '                                  "Opacity": 50' +
    '                                },' +
    '                                {' +
    '                                  "Text": "Örnek Metin 1",' +
    '                                  "Fontfilename": "Verdana.ttf",' +
    '                                  "FontSize": 14.0,' +
    '                                  "FontColorR": 0,' +
    '                                  "FontColorG": 0,' +
    '                                  "FontColorB": 255,' +
    '                                  "AddTo": 1,' +
    '                                  "LocationLeft": 10.0,' +
    '                                  "LocationBottom": 400.0,' +
    '                                  "Opacity": 80' +
    '                                }' +
    '                              ]';


let tamgaESRequest = new TamgaESRequest(params);

tamgaESRequest.on("message", function (tamgaESResponse) {

   //İmzalanan dosya base64 verisini yazdır
   console.info(e.signed_base64);

    //Uyarılar. Örn. CAdES-XL istenmesine rağmen zaman damgası parametresi boş  vb.
    for (let i = 0; i < tamgaESResponse.warnings.length; i++) {
        console.warn("WARNING : " + tamgaESResponse.warnings[0]);
```

```
    }

});


//Hata event i
tamgaESRequest.on("error", function (tamgaes_error_code) {
    alert(tamgaes_error_code);
});

//İsteği gönder
tamgaESRequest.run();
```